CHAPTER FIVE

Musicat Listens to Bad Melodies

This section of my dissertation (Chapters 5, 6, and 7) shows Musicat in action as it "listens" to various melodies. Recall from Chapter 1 the concept of "listening as performance". If this were a thesis on, say, a computer program that attempted to *play* Chopin nocturnes on a digital piano, this chapter would discuss the computer's "performance" in the most usual sense of the word; that is, it would mean using an instrument to make music for an audience. However, this is a thesis on Musicat, and Musicat's "performances" do not involve playing an instrument, but instead the dynamic act of *listening*. Listening is a temporal process that is hard to depict in static figures, and watching the program run is the ideal way to understand Musicat's *listening performance* and to see how the program's activity of listening, far from being passive, involves much work.

These three chapters give many examples of Musicat in the process of listening, using screen captures, or screenshots, to show Musicat's internal representations of the music it is listening to. Early in this chapter I use many screenshots to give a crude approximation to a watching a movie of the program running. These "movies" are crude indeed — they show just one frame for each measure of music, whereas Musicat's internal state may change not once but one *thousand* times per measure. Still, these crude "movies" should help the reader

understand how Musicat listens and how its internal representations (groups, analogies, and so forth) change over time as new melody notes are "heard" by the program. Later in this and followings chapters, to be more concise, I show nothing but the final state of the program after a run, even though this leaves out many important details of Musicat's performances.

In this chapter I first explain how to interpret the figures generated by Musicat, and give a detailed example of its listening performance for the simple melody "Twinkle, Twinkle, Little Star" (this melody is re-examined in Chapter 6). Next, I show Musicat listening to melodies from three different categories: *Bad* Melodies (this chapter), *Simple* Melodies (Chapter 6), and *Complex* Melodies (Chapter 7).

The first category, *Bad Melodies*, is comprised of melodies composed on purpose (or generated randomly) to sound very "unmelodic" and to lack any sense of large-scale structure. Why should we subject Musicat to bad melodies? For one thing, the bad melodies are intended to act as a foil to the idea that a trivial program would perform just as well as Musicat. We show that Musicat behaves exceedingly differently depending on whether or not a melody has an interesting structure.

Consider the following source of concern: Musicat is biased towards hearing music as composed of regular binary structures; that is, it has a default expectation that the first two measures will form a group, as will measures 3–4, and then those two groups will be grouped together, and then the next four measures will show the same pattern, and this group will then join with the first four measures, resulting in a larger eight-measure group, and so forth, in ever-increasing powers of two. Analogies between groups, similarly, often show a regular structure where the left-hand side of each group is mapped onto the right-hand side of the group. One often (but certainly not always!) hears this sort of binary structure in simple melodies. One might wonder if a trivial program that understood nothing about music but

that *always* generated a regular binary structure of this type would perform just as well as Musicat.

All the Bad Melodies introduced in this chapter lack such structure. If Musicat were consistently to hear strong, regular, binary structures in these melodies, that would be most troubling. However, we show that Musicat is confused by the lack of structure in these melodies (as a human would be): it flails around trying to form coherent internal representations, and it generally fails to find strong groups and analogies.

The next category of melodies, *Simple Melodies,* consists of typical examples of melodies in Musicat's domain. They are rather simply-constructed melodies that should be easy for humans to make sense of. I investigate the types and strengths of structures that Musicat forms for these melodies, and show how its listening performances are different for these than for the bad melodies.

The final examples of this chapter are the *Complex Melodies.* These are not "complex" with respect to music in general, but they are certainly more complex than those in the Simple Melodies category. Complex melodies involve greater variation in structure and not as much literal repetition of melodic motives. They are also longer (that is, they may be 32 bars long instead or 8 or 16). We show how Musicat finds some interesting structure in these melodies, but also has more trouble understanding the complexity.

Understanding Musicat's Listening Performances

Musicat employs a unique graphical notation to display its internal representations of musical structure. Figure 1 shows an example, with labels identifying the different parts:

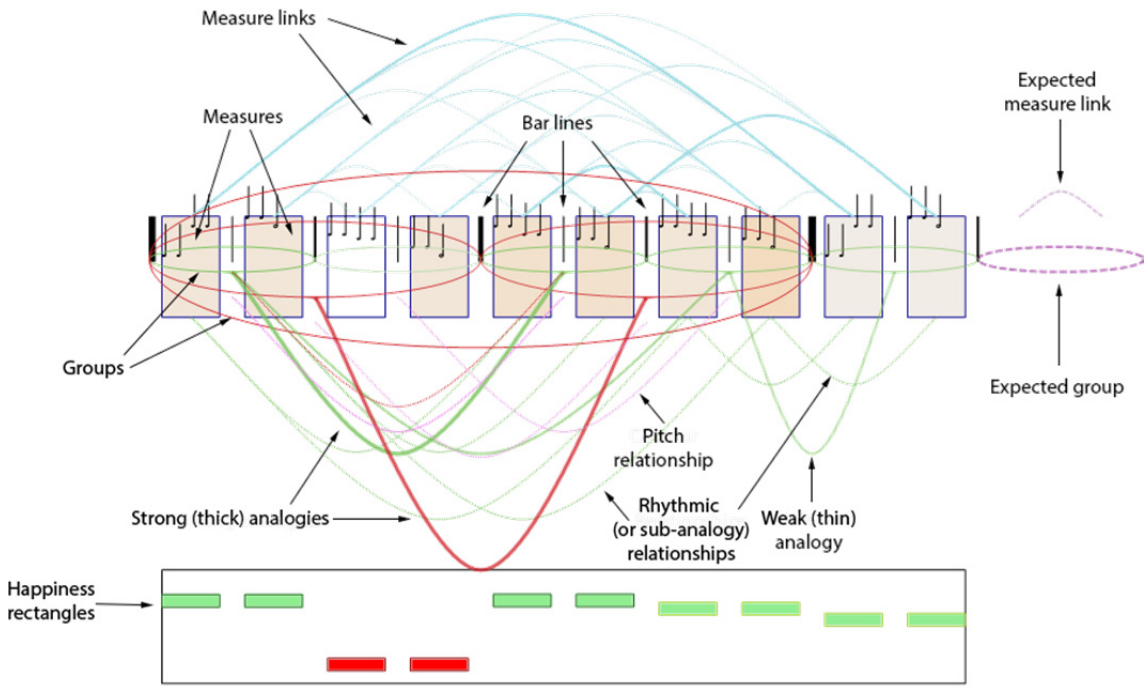

Figure 5.1: A sample screenshot demonstrating Musicat's notation.

The figure is divided into two major parts: music and associated structures on top, and a display of measure "happinesses" on the bottom. Let's start with this simpler bottom part: it consists of a series of rectangles where the height (and, redundantly, the color) of each rectangle indicates the computed *happiness* of the measure directly above it. Happiness is described in more detail in Chapter 7. For now it is sufficient to understand that if the program has discovered good, strong structures associated with a measure, the measure is assigned high happiness; weak structures or missing structure (for instance, a measure that has not been assigned to a group) will have low happiness. In Musicat, happiness is inversely related to *temperature*, a notion borrowed from other FARG programs such as Copycat. The presence of temperature is most evident in the figure for high-temperature cases: low-happiness measures have high temperature, which is indicated by the red color of their

"happiness rectangles", directly below them. (Low-temperature measures have green happiness rectangles below them, to emphasize high happiness, not low temperature.)

The top part of the figure displays the melody the program has heard so far, using a very simplified music notation, devoid even of staff lines. Each measure is represented by a rectangle containing the notes of the measure. Bar lines are drawn between the measures. Unlike in traditional music notation, these bar lines have variable thickness. A thick bar line indicates that the program expects the following measure to be heard as having a strong hypermetric accent, and thus that the bar line is a likely spot for a group boundary to form.

Each measure is shaded with a color corresponding to the amount of *attention* the program has given that measure in a very short recent time window. When the program is running, the measures flicker very noticeably as attention shifts between different measures.

Blue arcs are drawn at the top of the figure, above the music, connecting pairs of individual measures. These connections are called *rhythmic measure-links*, or simply *measure links*. They are formed when the program detects that two measures are similar to each other in terms of rhythm — pitch is ignored in the creation of these links. The strength of similarity is indicated by the thickness of the blue arc. In Figure 1, there are lots of arcs, which is unsurprising because each measure of this melody is one of only two possible rhythms — four quarter notes (QQQQ) or two quarters followed by a half note (QQH).

Measures can be grouped together. *Groups* are indicated by ovals enclosing (or superimposed upon) the measures in question. Groups may contain not only measures, but also other groups. Such groups (occasionally called *meta-groups* to avoid ambiguity) also are indicated by ovals enclosing the contained groups. The color of an oval varies with the size and strength of a group. The color also depends on context, and is chosen by the program to help make stronger and larger groups stand out. Note that there is no *a priori*, context-free meaning of group color. Typically, smaller groups are colored green and larger groups are

colored red, but as the program runs, a group that used to be colored red might turn green, or vice versa. This reflects nothing except for changing context. The thickness of an oval indicates group strength, as does opacity; for example, a weak group will be indicated with a thin and partly transparent oval.

The arcs drawn below the measures of music can be less easy to interpret because such arcs can mean several different things. The type of arc easiest to understand is a large arc attached on each end to a group: this is an *analogy*. For example, the large red arc in Figure 1 indicates an analogy between the group of measures 1–4 and the group of measures 5–9. Also notice that the groups linked by the analogy are both colored red. This is intentional: groups and analogy-arcs are chosen to be the same color whenever possible.

For each analogy, additional thin arcs are drawn using a muted version of the color used to draw the analogy. These arcs connect pairs of analogy components, in order to indicate the mapping that makes up the analogy; these arcs correspond to discovered *relationships* between components on either side of the analogy. Relationships can be divided into three categories: pitch relationships, rhythmic relationships, and analogy relationships. *Pitch relationships* (sometimes called "non-rhythmic relationships") are based on pitch features such as contour or melodic tension. Pitch relationships are drawn with pink color (instead of the muted analogy-color) to distinguish them from the other two relationship types. *Rhythmic relationships,* the most commonly-appearing relationship type visible in the figures, indicate a relationship based on rhythmic features. *Analogy relationships* are indistinguishable in the figures from rhythmic relationships (when considering the drawing color alone), but they are not very important to the reader and I mention them here just for completeness: they indicate that an analogy has been chunked in order to be used as a component of another analogy. This is a bit redundant in the figure, because the original

analogy is also visible, but this type of relationship was necessary in the program for technical reasons. I the rest of this chapter and the following two chapters, for brevity, I discuss Musicat's relationships as if there were only two types: pitch and rhythmic; analogies are discussed separately.

Unfortunately, in the screenshots that follow it is hard to distinguish between pink arcs indicating pitch relationships and light-red arcs indicating rhythmic relationships for an analogy that is colored red. The pitch relationships were drawn with a finely-dotted texture in an attempt to distinguish them further, but this texture is also hard to discern. One clue is that most of the rhythmic relationships are drawn such that they connect with individual measure boxes, whereas the non-rhythmic relationships have endpoints that are near the groups involved and vertically positioned higher in the figure than the rhythmic relationships. I will comment on the type of these potentially ambiguous-looking relationships in the text when necessary.

Finally, at the right-hand side of the screen, we sometimes see structures indicating Musicat's *expectations*. Any expected structure is drawn using dashed purple lines. In Figure 1, the purple oval at the right indicates that the program expects a two-measure group to form in that location. Additionally, the purple arc above indicates that it expects a measure link to form between the two measures of that group.

Some Simplified Notation

For simplicity of notation, in the rest of this thesis, a group around measures 1–4 may be written with parentheses and in boldface as follows:

(1–4)

Even if a group such as this one contains subgroups, the group is still written as above, using just the measure numbers of the start and the end of the group; I generally avoid notation such as **((1–2)(3–4))** and prefer instead to leave the subgroups implied, to reduce clutter in the text. This is especially helpful in discussing analogies: using this notation I can write about an analogy between groups **(1–4)** and **(5–8)**. Analogies also have a special notation and may be written with a double-sided arrow between groups:

(1–4) ↔ (5–8)

When referring to a range of measures that may or may not involve a group, I simply write the range of measures without boldface or parentheses:

1–4

A Note on the Pronouns "I" and "We"

Most of this thesis is written in the first person singular as I (Eric) describe my work. However, the pronoun "I" would be inaccurate in many places because I actually developed Musicat in close collaboration with my advisor, Douglas Hofstadter, over the course of many years in Bloomington and six months in Paris (or, as he would say, in "Villefleury" and "Lutèce"). Thus, there are places in this text where I write "we" instead of "I" to emphasize ideas or observations that are a result of this collaboration. This is especially apparent in this chapter and the two following ones, in which I slip back and forth between describing my own observations of Musicat in action and observations that were more collaborative. Thus, "I" refers to Eric, "we" refers to both of us, and the pronoun "one" is used where necessary to

discuss a general, abstract person (often "we" would be used for this latter purpose, but here "we" refers to two specific people). However, there are also some places where I find it natural to involve you, the reader, in my observations, writing phrases such as "we see in the figure that such-and-such has happened". It should be clear from context when I am using "we" in this manner.

First Example: Twinkle, Twinkle, Little Star

Before jumping into examples of Bad Melodies, I present an example of Musicat listening to the familiar tune "Twinkle, Twinkle, Little Star", which will appear again in the next chapter (this is an example of the Simple Melody category, even though the rest of this chapter deals with Bad Melodies). Remember that Musicat "listens" to music presented not as an audio recording, but rather as a series of symbols such as would be found in sheet music. We can approximate what Musicat receives as input if we imagine a sheet of paper covering up the entirety of Figure 2 below (the "Twinkle, Twinkle" melody) and then allow this imagined paper to slide slowly to the right so that it uncovers the notes one at a time, with the speed of sliding corresponding to the duration of each note.

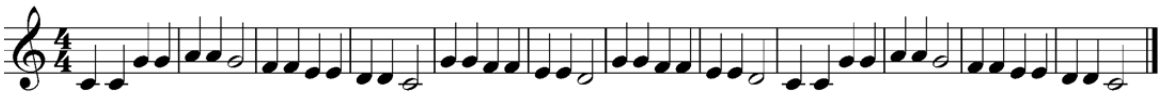

Figure 5.2: Twinkle, Twinkle, Little Star.

Human listeners who already know this melody might remember the lyrics as well. In the rest of this chapter I present melodies along with their lyrics, simply as a memory aid for the reader — words make it easier to recall tunes. But keep in mind that the *program* has no knowledge of the lyrics to any of these songs.

Without further ado, let us watch Musicat "listening" to "Twinkle, Twinkle".

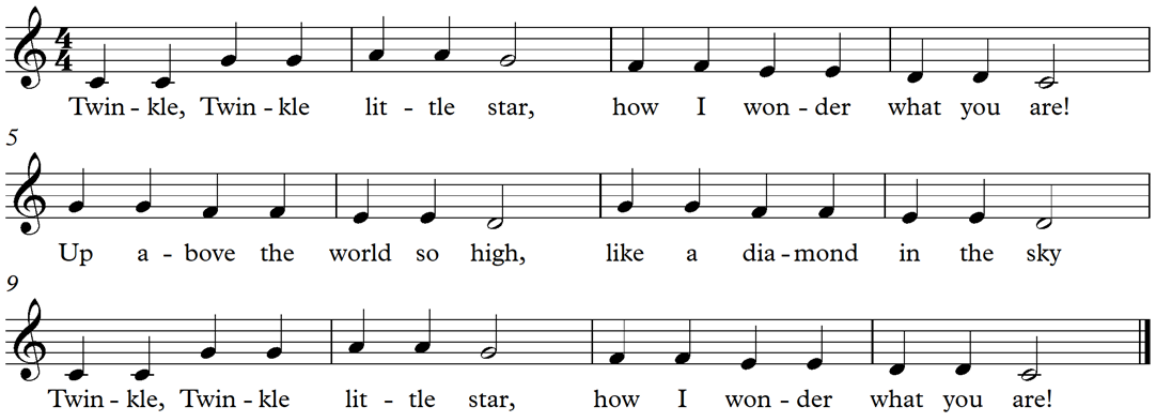

Figure 5.3: Twinkle, Twinkle, with lyrics.

To start the listening session, the user simply starts the Musicat program, selects a melody from a list (or types the melody in using a special notation to indicate the notes and rhythms), and presses "Go". The notes of the melody ("Twinkle, Twinkle", in this example), start appearing on the screen and the program starts "listening". The rate at which successive notes appear on the screen is roughly proportional to the musical duration of each note (the ratio being determined by the program's current speed setting). For example, the first seven notes of "Twinkle, Twinkle" will appear with approximately the same amount of time in between each successive pair of notes. However, the seventh note (G, on "star") is a half note, so if one second was spent on each quarter note earlier, it will take two more seconds until the eighth note (F, on "how") appears.

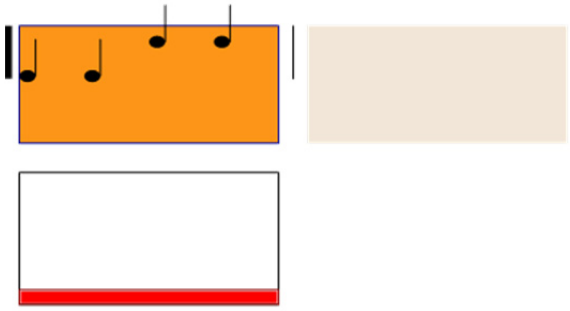

Figure 5.4: Twinkle, Twinkle, measure 1.

The figure above shows Musicat's display after the first complete measure has been presented. Because this version of Musicat is restricted to forming groups at least one measure long (and with a length that is an integral number of measures), no groups have formed inside this measure. (Otherwise, we would expect each pair of repeated notes to form a small group.) No groups or analogies have formed, but some other things have happened. A *thick* bar line has been drawn before the measure, to indicate that this measure has been heard as having a strong hypermetric downbeat. A *thin* bar line has been drawn after the measure, indicating Musicat's guess that measure 2 will probably be heard as being in a weak hypermetric position. The bar lines, incidentally, are drawn only in the top half of the space between measures, simply in order to de-clutter the display. The height does not represent anything, just as with a normal bar line.

The first measure has been shaded dark orange, to show that Musicat has focused nearly all of its attention on this measure. A small amount of the program's attention has been devoted to considering expectations about the second measure, as is indicated by the lighter-colored shaded rectangle to the right, in the spot where measure 2 will occur. The happiness of the first measure has been computed, resulting in the lowest value possible, because the measure is not part of any group.

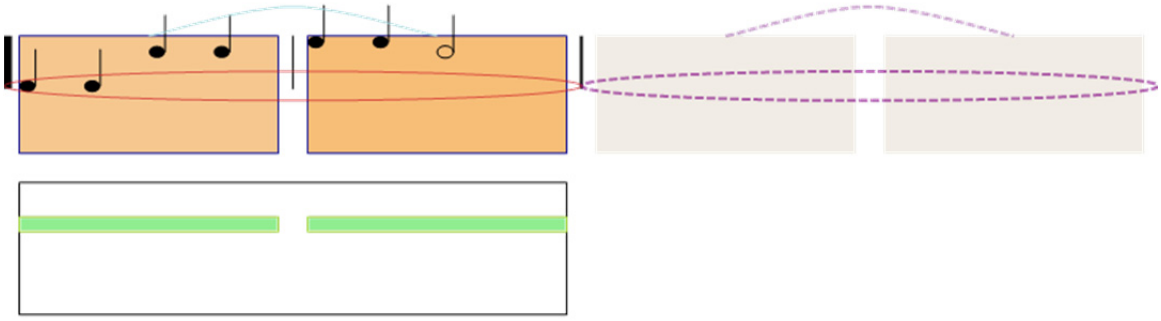

Figure 5.5: Twinkle, Twinkle, measure 2.

After measure 2 has been heard, several important things have happened: a link has been found between these two measures, because they have a similar rhythm. A group has also formed around measures 1–2, and an expectation has formed for another 2-measure group to follow. The first two measures also have relatively high happiness scores. Finally, we can see a bar line after measure 2 that is slightly thicker than the one preceding that measure.

Musicat's expectations (in the present version of the program) are not the melodic expectations for particular *pitches* to occur next, as in Larson's or Margulis's computer models; instead, they are expectations for particular *structures* to occur next. As will become obvious, Musicat's expectations are rather simple-minded; ideally, structural and pitch-based expectations would both occur in the model and would reinforce each other.

One important feature of Musicat's expectations is not shown explicitly in the figures, simply in order to de-clutter the display: when Musicat draws a purple expectation, it also creates an expectation (hidden in the figures) for an analogy to form later, linking the expected group to the group from which it derives. In the present example, Musicat expects another 2-measure group, **(3–4)**, to form later on, and also expects an analogy to form between the red group and the future group **(3–4)**.

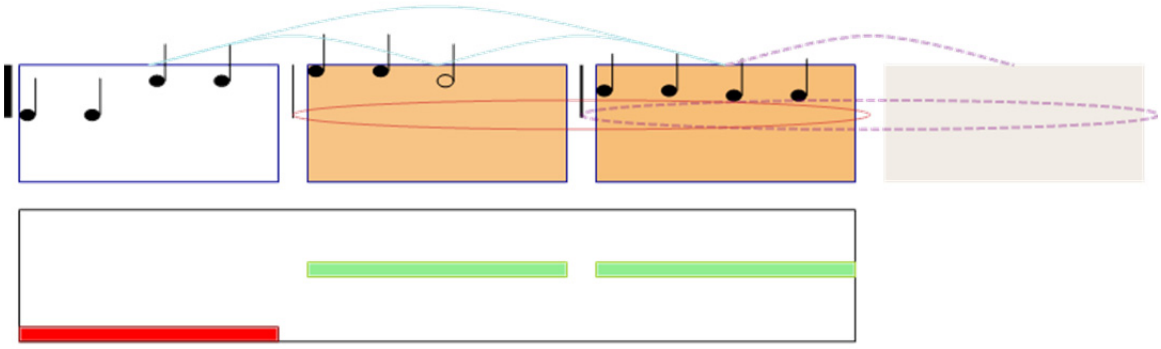

Figure 5.6: Twinkle, Twinkle, measure 3.

After measure three, the picture looks confusing. Group **(1–2)** has unfortunately disappeared, and instead measures 2–3 have (also unfortunately) been grouped together. This is unfortunate, because we want Musicat to quickly and easily hear measures 1–2 as a group, and to not make the mistake of hearing such an unlikely-seeming pair as measures 2–3 as a group (for a human listener, the half note G in measure 2 is enough, in this context, to make the grouping structure quite obvious.) However, even though the creation of group **(2–3)** is disappointing for the human outside observer at the instant in time captured by the figure, it is nevertheless an expected and necessary consequence of Musicat's architecture. The program changes its mind all the time, and any appearance or disappearance of a group, good or bad, is subject to reversal. Even though it looks as if just one group has been destroyed and another created (when we look at the difference between this figure and the previous), in reality many different group creation and destruction events might have occurred between the times that these snapshots were taken. Furthermore, the program's creation and destruction of groups and other structures occurs in totally different ways on different runs, because of its nondeterministic or stochastic architecture. Thus, one should not be too surprised when groups such as **(2–3)** are formed — indeed, in some contexts such flexibility is absolutely necessary — but one should still hope that Musicat will quickly reverse any unnatural-looking decisions and settle on a human-like hearing.

Several other things have changed in the figure, compared with the previous one. Additional measure links have formed between all pairs of measures. The happiness of measures 2 and 3 is moderate, but the happiness of measure 1 is now zero; the red bar indicates low happiness (and hence high temperature). The program has spent most of its energy examining measures 2 and 3 (as indicated by the orange shading), but the low happiness should help it direct its attention back to the first measure.

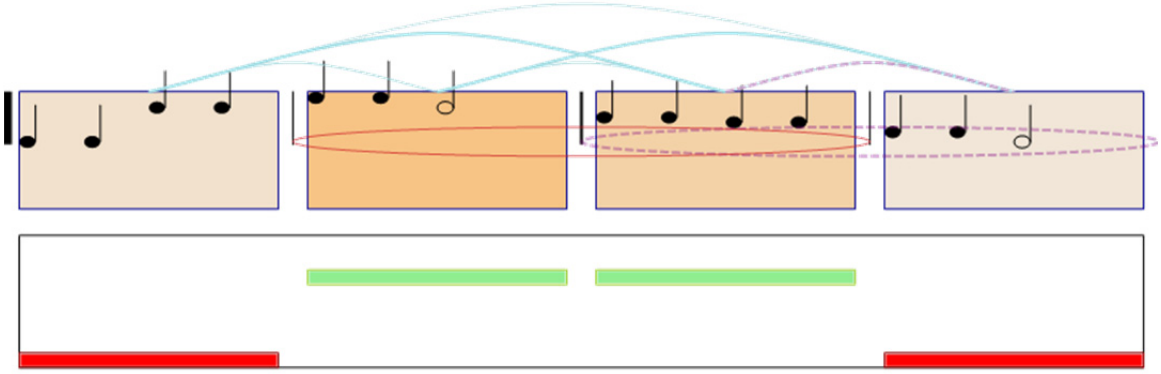

Figure 5.7: Twinkle, Twinkle, measure 4.

After measure 4 has arrived, the same strange group **(2–3)** persists, as does the expectation for the group **(3–4)**. The happiness score for measure 4, as for measure 1, is zero. More measure links (based solely on rhythm) have been formed. In this picture we can see how measures 1 and 3 are strongly linked (after all, they have identical rhythms — four quarter notes in each), as are measures 2 and 4, also because their rhythms match. Another thin bar line has formed after measure 3.

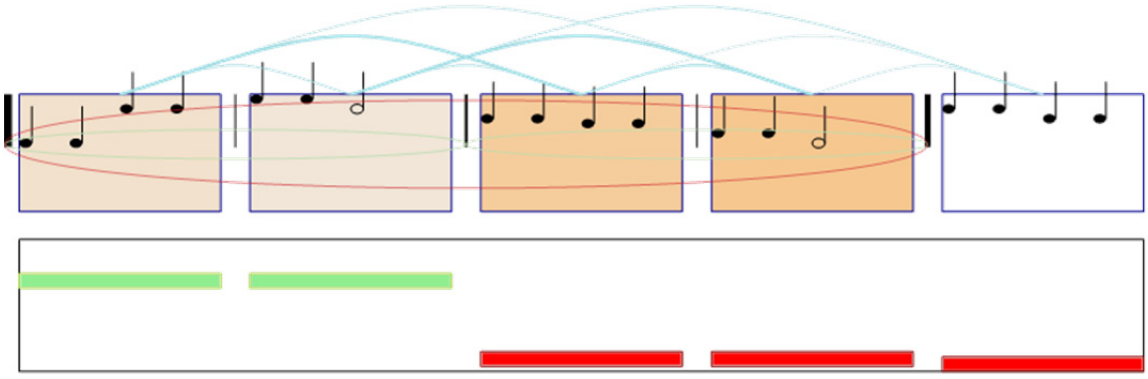

Figure 5.8: Twinkle, Twinkle, measure 5.

Suddenly, after measure 5 has been heard, the grouping structure has changed radically. Measures 1–2 as well as 3–4 form (light blue) groups, although they are difficult to see in the figure — you may have to look hard to see them! A larger-scale meta-group (red) has formed around those two groups. The strange group **(2–3)** has disappeared (fortunately). A thick bar line has formed after measure 4. The first two measures have high happiness scores again, although the program is unhappy about the rest of the measures, because group **(3–4)** has been perceived as very weak and measure 5 is not involved in any group at all.

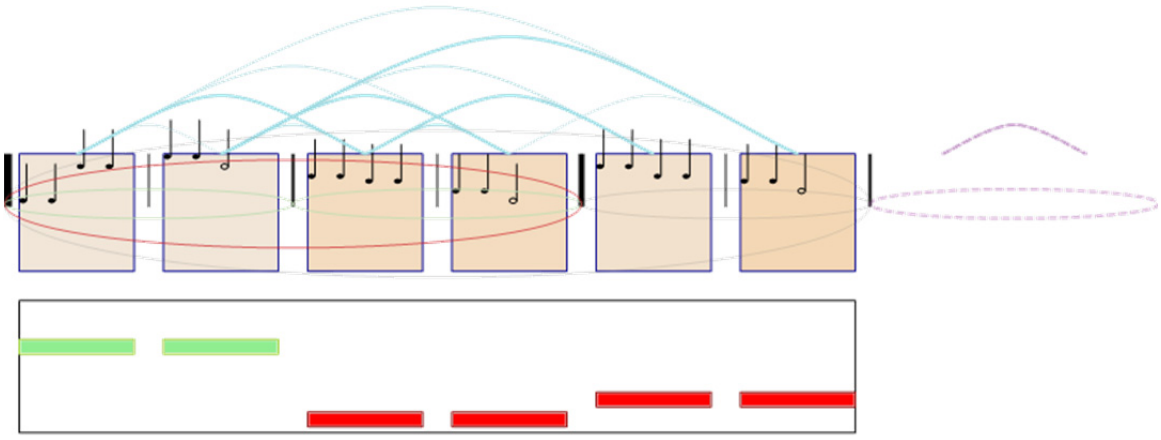

Figure 5.9: Twinkle, Twinkle, measure 6.

After measure 6 has been heard, an additional group has formed, **(5–6)**, and it has been incorporated into a very weak (and thus barely-visible) meta-group, **(1–6)**, encompassing all the measures heard so far. Another expectation has formed, this time for the 2-measure group **(7–8)**.

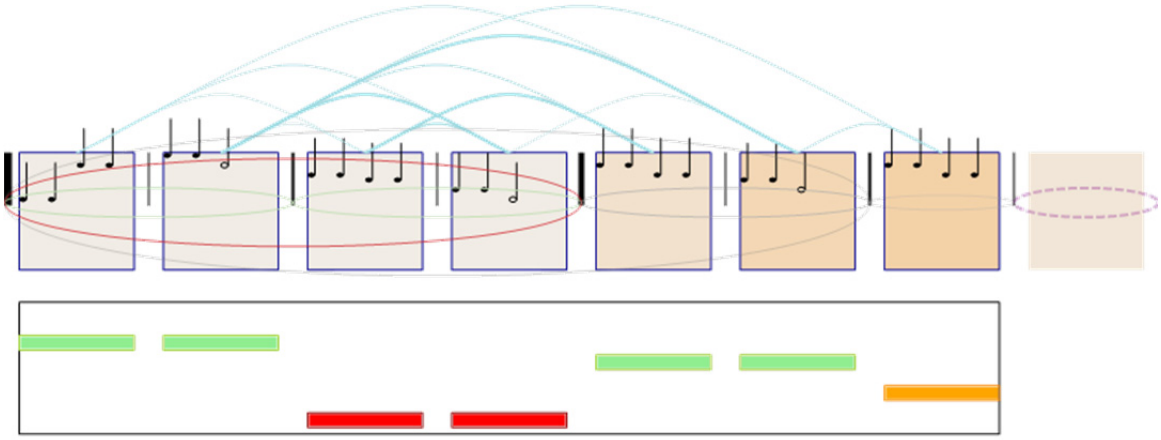

Figure 5.10: Twinkle, Twinkle, measure 7.

Here, after measure 7 has been heard, the structure formed looks similar to that in the preceding figure, although the happiness values for measures 5–6 have increased, and a weak 1-measure group has formed that contains the most recent measure. The observant reader may wonder why measures 3–4 have such low associated happiness values, while measures 5–6 have improved in happiness. The answer is, in part, that group **(5–6)** is now contained in group **(1–6)**. However, this isn't enough of an explanation, since group **(3–4)** is also contained in a parent group, **(1–4)**. The full story is that as Musicat runs, it builds up a list of qualities of each group, and this list is used to compute group strength. (I call this the list of "Group Reasons" in a later chapter, but it refers to the same thing.) Like everything else in the program, this list is created in a stochastic manner, with different group qualities being noticed in different orders on each run. In this run, Musicat either has not noticed

some of the strong qualities of group **(3–4)**, or else it has perceived some negative quality that makes it think it is a very weak group. While the program is running, I can pause it and display the list of group qualities, but in most cases I have let the program run without stopping to record these details. Calculation of group strength (and the associated measure happinesses) is an important topic, but this section of my thesis is focused on Musicat's behavior instead of the lowest-level mechanisms underlying the behavior — see Chapter 9 for the details. In the present example, I would indeed like to see what happened to group **(3–4)**, but I can't simply re-run the program because the program is stochastic and I neglected to record the random seed used for the run; moreover, I made some slight changes in the code after this particular run, so now the behavior would be slightly different, even if I had access to the random seed.

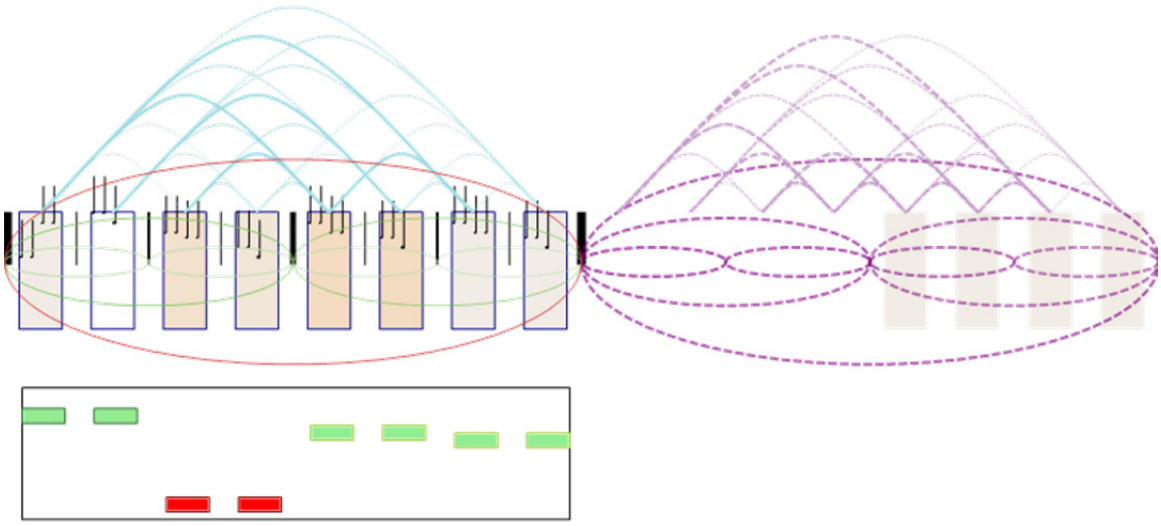

Figure 5.11: Twinkle, Twinkle, measure 8.

Now, after measure 8, another dramatic change has occurred: all of the first 8 measures have formed very regular duple structures. The program is happy with all the measures except for 3–4; for some reason, it still considers these to form a weak group, and these two measures will remain unhappy for the duration of the run. Additionally, a very

large and complex expectation has formed for another entire 8-measure structure like this one to follow (note that "Twinkle, Twinkle" only has 12 measures, so this expectation obviously will not be fulfilled).

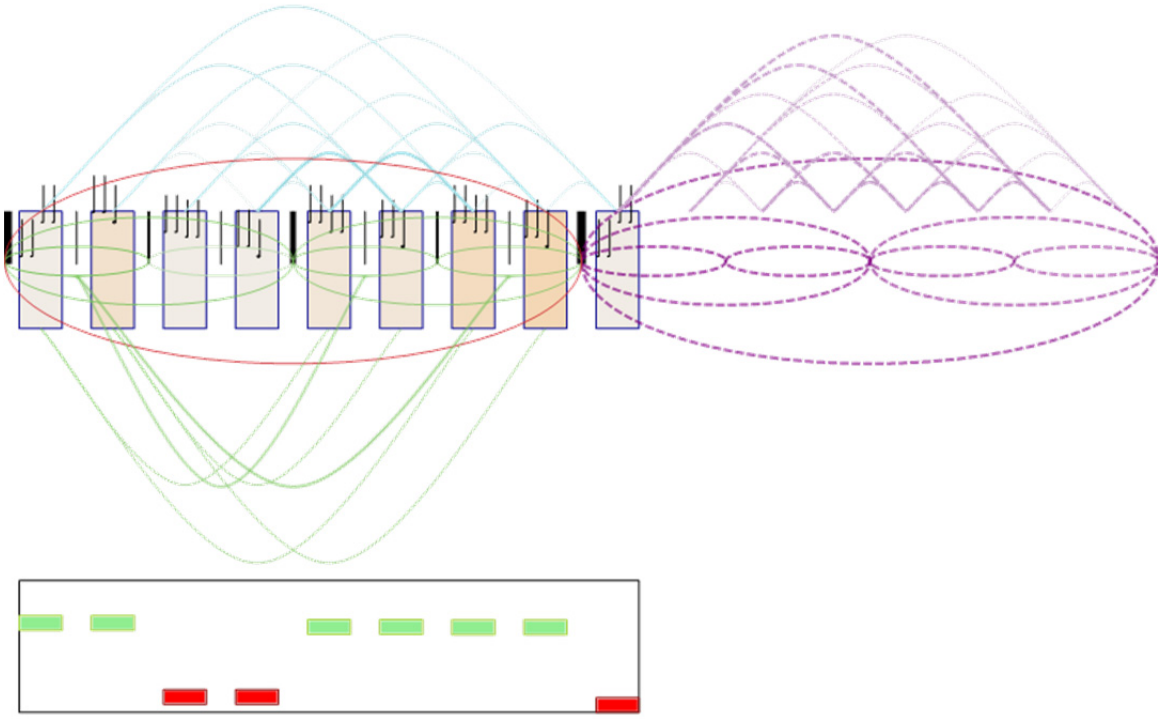

Figure 5.12: Twinkle, Twinkle, measure 9.

Suddenly, two analogies have formed! Both involve the group **(1–2)**: one makes a link between **(1–2)** and **(5–6)**, while the other makes a link between **(1–2)** and **(7–8)**. It should be easy to see that measure 2 is similar in shape to measures 6 and 8, and also that all three of the groups involved, **(1–2)**, **(5–6)**, and **(7–8)**, have the same rhythm (QQQQ QQH). Thus, these analogies should make intuitive sense. However, one obvious similarity in the music has not been noticed by the program: **(5–6)** and **(7–8)** are identical! The most obvious analogy in this melody is the trivial mapping **(5–6)↔(7–8)**, but sadly Musicat has

not noticed it. This oversight is very weird, but don't lose hope yet — perhaps the analogy will be formed later in this run.

Not much has changed in terms of grouping. The new measure has been heard, but not incorporated in any structure, so it has low happiness.

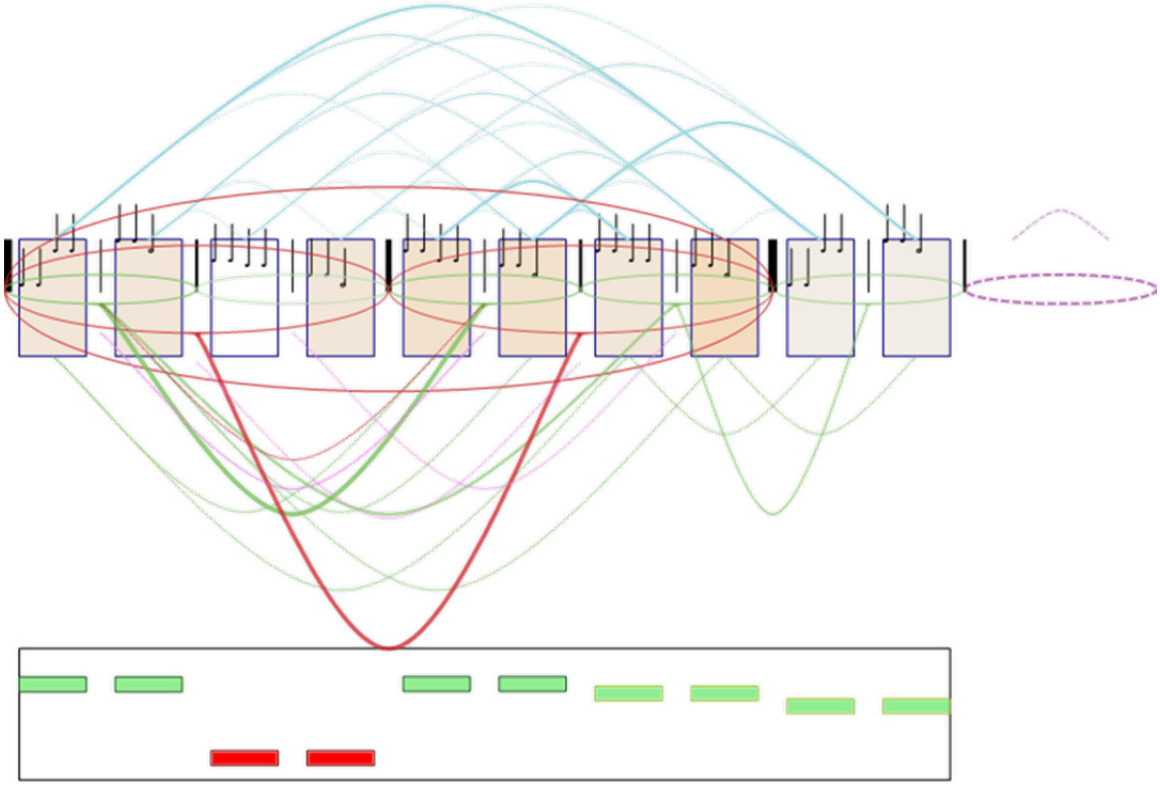

Figure 5.13: Twinkle, Twinkle, measure 10.

Now, just one measure after the first analogies appeared, a larger-scale analogy has formed between groups **(1–4)** and **(5–8)**. These two groups are colored red, matching the color of the analogy, to make it easier to see the structure. The analogy between groups **(1–2)** and **(5–6)** has been strengthened.

The analogy between **(5–6)** and **(7–8)** is still missing, however. Likewise, human listeners will notice another exact repetition that has occurred: measures 9–10 are a

recapitulation of measure 1–2; they are exactly the same and occur in a similar hypermetric position, so it is trivial for a person to hear the analogy **(1–2)↔(9–10)**. Notice that Musicat has created a strong measure link (based on rhythm) between measures 1 and 9, and a weak one between measures 2 and 10, which should give it a hint to look for an analogy involving those measures. Likewise, there is a strong measure link between measures 5 and 7, so it has noticed part of the similarity required to make the **(5–6)↔(7–8)** analogy. Will Musicat notice either of the missing analogies?

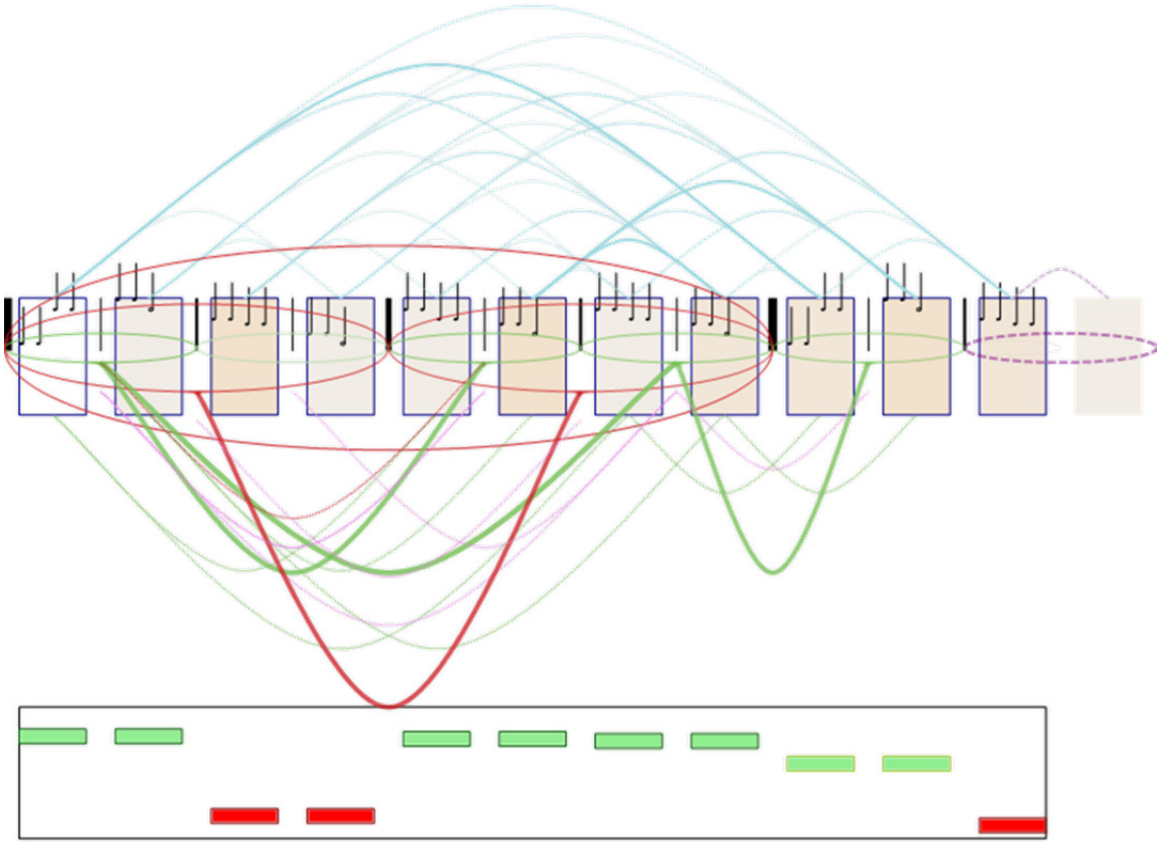

Figure 5.14: Twinkle, Twinkle, measure 11.

The same structures are in place as in the previous view, but some analogies have become stronger. Conversely, some of the measure links have weakened over time; notice how *recent* measures have long-distance links to earlier measures (such as the strong links above the measures linking measure 1 to 9 and measure 6 to 10), but there are not many links between pairs of *old* measures; those have mostly faded away.

Our two missing analogies are still missing, and the link between measures 5 and 7 has faded, so there is still some hope to see the **(1–2)↔(9–10)** analogy, but less for the other.

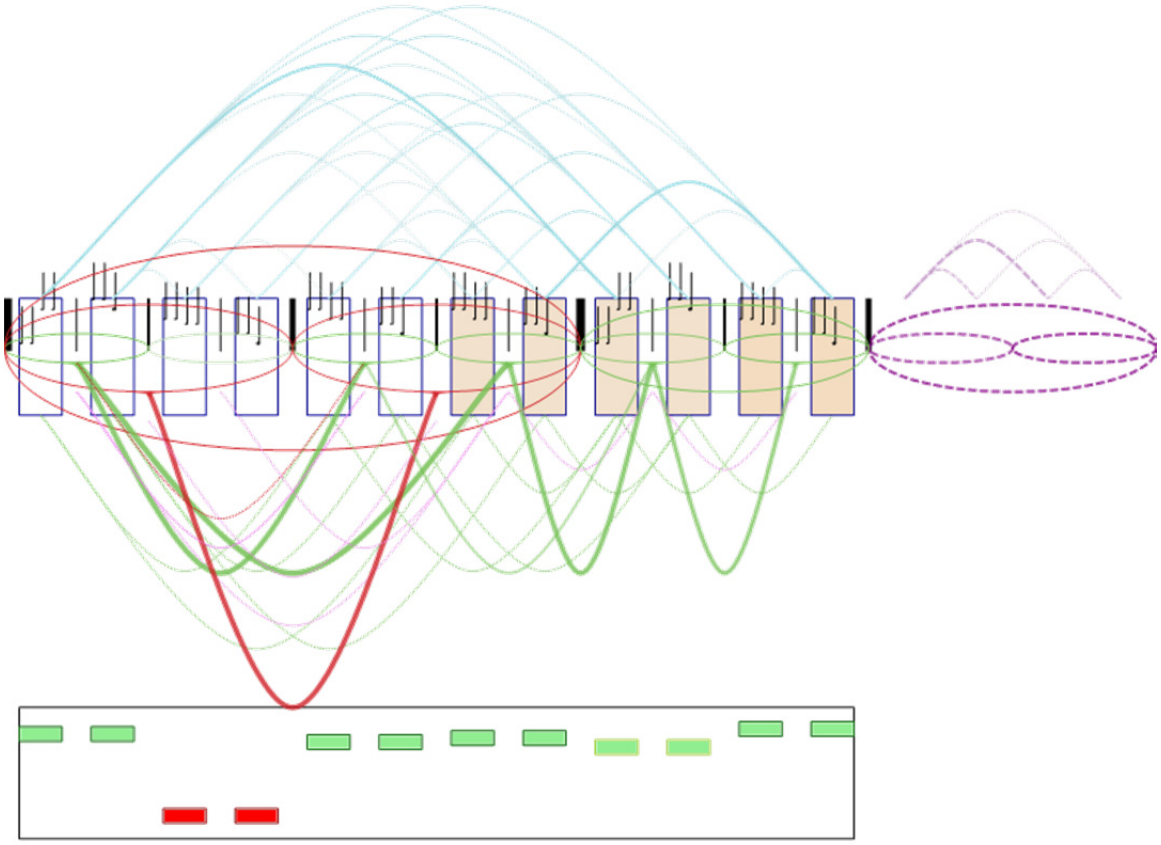

Figure 5.15: Twinkle, Twinkle, measure 12 (end of song).

The song has ended, but Musicat doesn't know it yet. The program has formed a new group, **(9–12)**, and expects another 4-measure structure to occur. This is not surprising, because 16-bar songs are extremely common. A new analogy at the end of the melody, **(9–10)↔(11–12)**, has been created.

The analogy **(5–6)↔(7–8)** still has not formed, and there is little reason to hope Musicat will notice it now, since its attention is focused on recent music. However, the blue measure link between measures 1 and 9 is still relatively strong, so there is still hope that the analogy **(1–2)↔(9–10)** will appear.

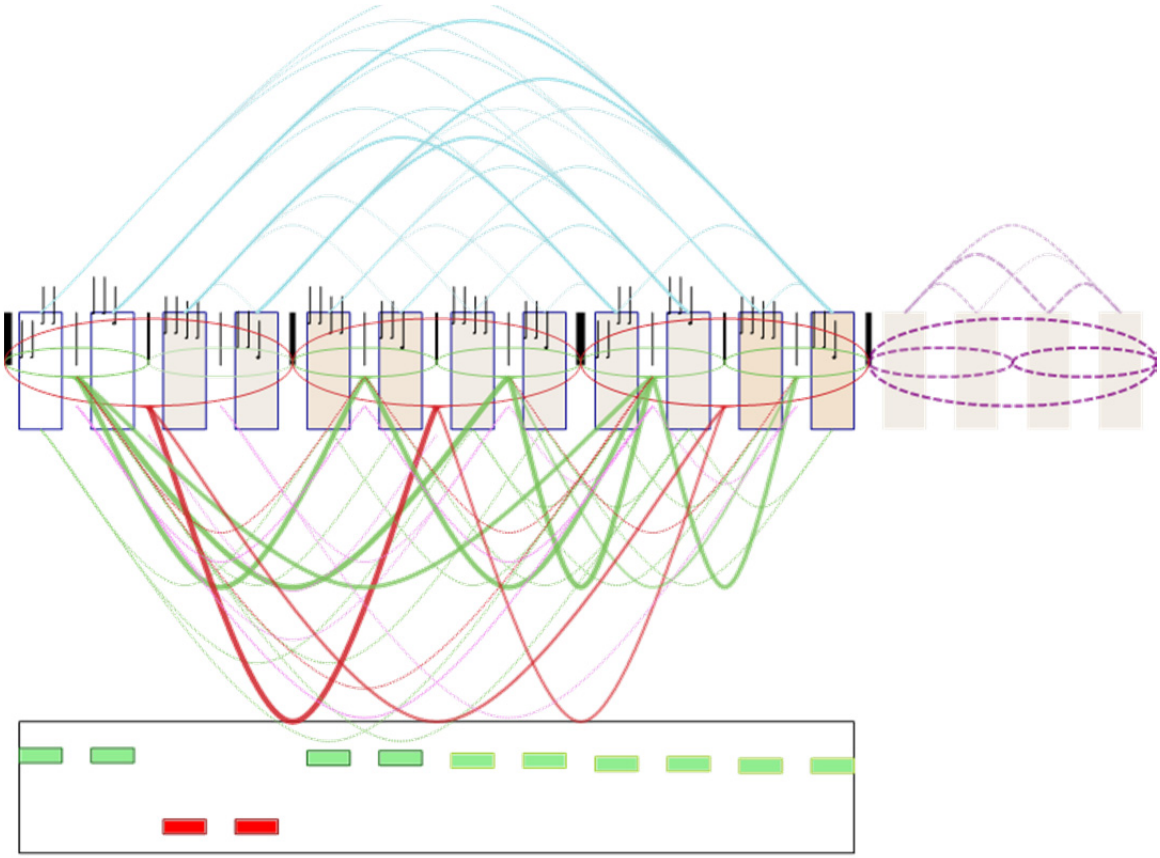

Figure 5.16: Twinkle, Twinkle, processing.

Processing continues for several measures' worth of time after the melody is complete. After some additional time, the analogy **(1–2)↔(9–10)** has formed at last! This is important because it was such an obvious analogy for a human listener, and also because these low-level analogies help pave the way for larger ones. Indeed, two more high-level analogies have formed. The new (thin red) analogy **(5–8)↔(9–12)** is weaker than the (thick red) analogy **(1–4)↔(5–8)**, but note that they are analogies each composed of essentially the same musical material, since the notes in measures 1–4 and 9–12 are identical. This similarity has been noticed by the program: a long-distance (thin red) analogy has formed between these two groups: **(1–4)↔(9–12)**, no doubt made possible by the **(1–2)↔(9–10)** analogy.

All three of these large analogies seem to be reasonable; it is easy to hear similarity (or identicality) in these pairs of groups.

In this large mass of analogies, it is disappointing that the analogy **(5–6)↔(7–8)** never formed; Musicat seems to have seen a great deal of the other interesting relationships in the melody. But each run is different, and in each run it pays attention to different features of the music.

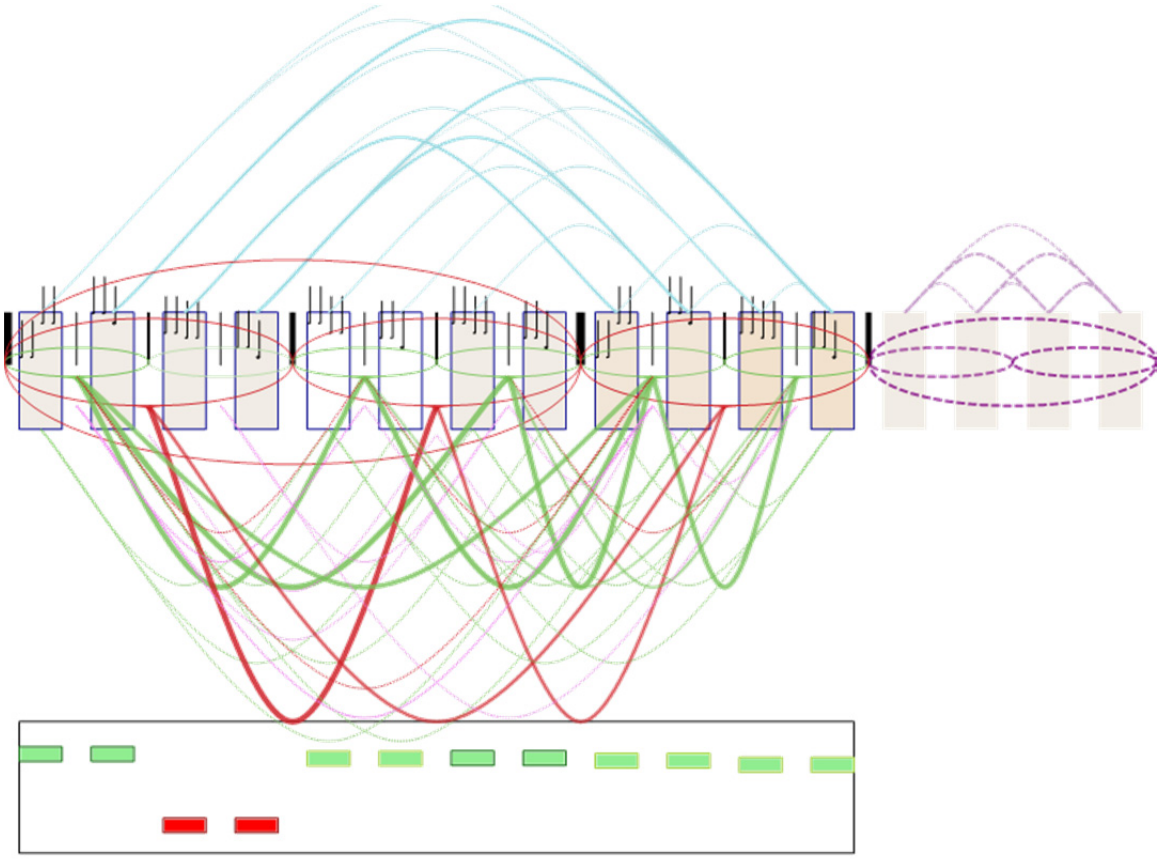

Figure 5.17: Twinkle, Twinkle, processing.

In the previous figure, Musicat is in its last moment of "listening" before the song is completely finished. The structures in the previous figure have been augmented with one more meta-group, **(1–8)**. Most of the measures, except 3–4, have high happiness.

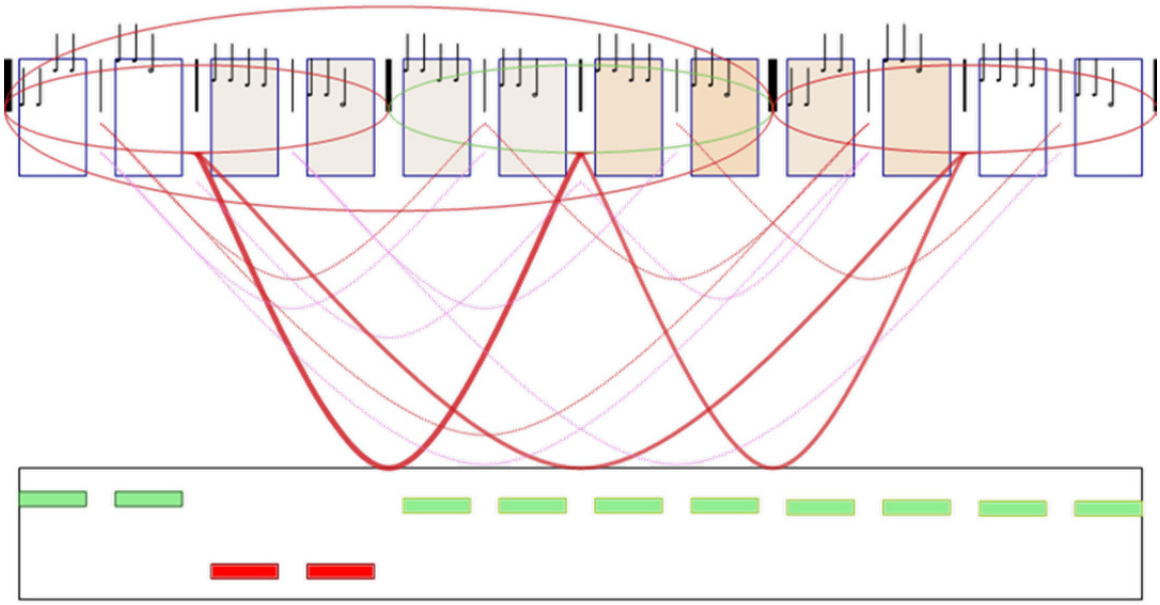

Figure 5.18: Twinkle, Twinkle, final state, low detail level (strongest structures only).

This figure shows the strongest elements remaining in the Workspace at the end of the run. The most important things are the three large-scale (red) analogies, the three 4-measure groups **(1–4)**, **(5–8)**, and **(9–12)**, and the larger 8-measure group **(1–8)**.

On this run, Musicat has heard the music in a fairly reasonable way. To human ears, this melody is an example of a simple **ABA** form; that is, we hear an opening 4-measure theme, a separate 4-measure long middle section, and then an exact repetition of the first theme. In this run, Musicat has noticed some of this structure; the relationship **(1–4)↔(9–12)** has been heard (although Musicat represents it as a strong *analogy*, not as a literal *repetition*). The middle section is in its own group **(5–8)**. But the program has also

heard a higher-level 8-measure grouping, so "**(AB)A**" would be a better description of its listening than simply "**ABA**". Additionally, it has heard myriad smaller connections between measures and groups. Interestingly, the analogies **(1–4)↔(5–8)** and **(5–8)↔(9–12)** indicate that the **A** and **B** sections are related; the mere description "**ABA**" fails to capture this relationship.

The most glaring problem in this run was the program's "deafness" to the similarity between **(5–6)** and **(7–8)**; these identical measures should have been heard as (trivially) analogous. But fortunately, on many other runs with the same melody it does notice this analogy; for example, see the next figure, showing the end of another run.

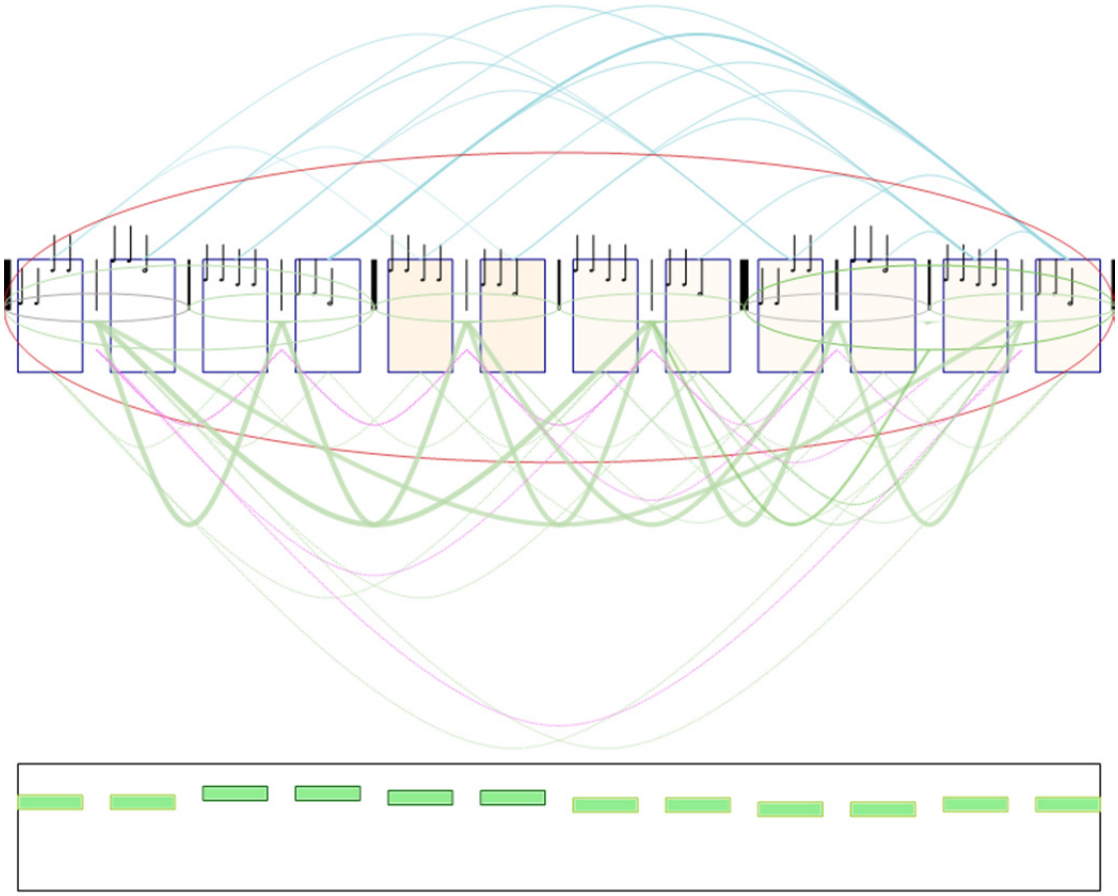

Figure 5.19: Another run on "Twinkle, Twinkle".

Bad Melodies

In the remainder of this chapter I present five “bad melodies”. For each melody I describe how it was constructed and then show Musicat listening to the melody. Because the program is stochastic and has different behavior each time it listens, I give examples from multiple runs for some melodies, to illustrate some of the possible variations in listening behavior.

BAD MELODY #1

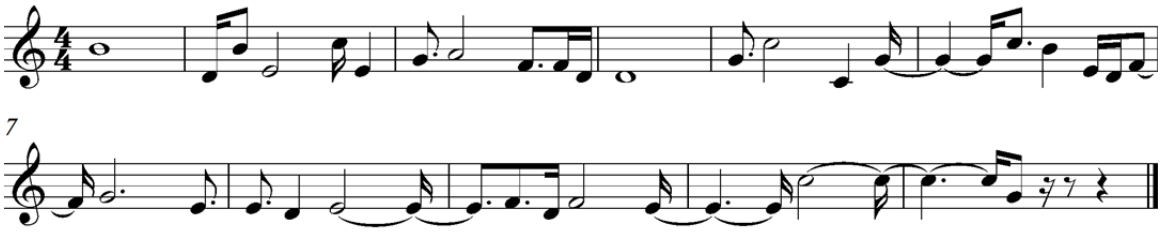

Figure 5.20: Bad Melody #1 (random rhythms and notes).

I wrote a simple computer program to generate this melody using a pseudorandom number generator. The program always generates 32 notes. For each note, the pitch is chosen randomly from the white keys of the piano between middle C and the next C an octave higher. The duration for each pitch is selected at random from the following list: whole, half, quarter, eighth, sixteenth, dotted half, dotted quarter, or dotted eighth. (Ties are visible in the notated music as a result of notes starting in strange places.) No rests are allowed (except at the end of the final measure). The time signature is fixed to 4/4.

The melody sounds just as random and unstructured as one would expect, given the algorithm that generated it. I find the notes confusing and unmemorable — it indeed sounds like a series of random notes, undeserving of the term "melody". So what happens when Musicat listens to it?

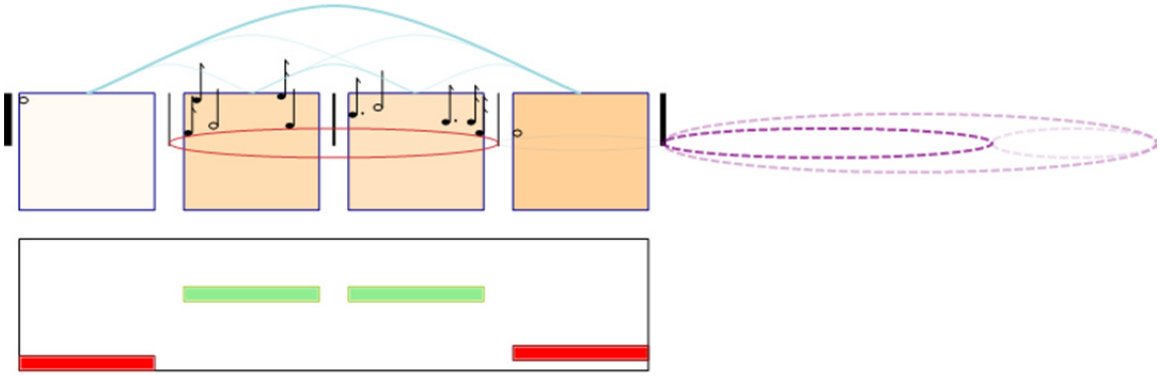

Figure 5.21: Bad Melody #1, measure 4.

The figure above shows the program's state after measure 4. This picture looks very similar to the one after measure 4 in the "Twinkle, Twinkle" run above: a group **(2–3)** has formed, leaving out measures 1 and 4. However, the program has noticed a link between measures 1 and 4: they are both whole notes.

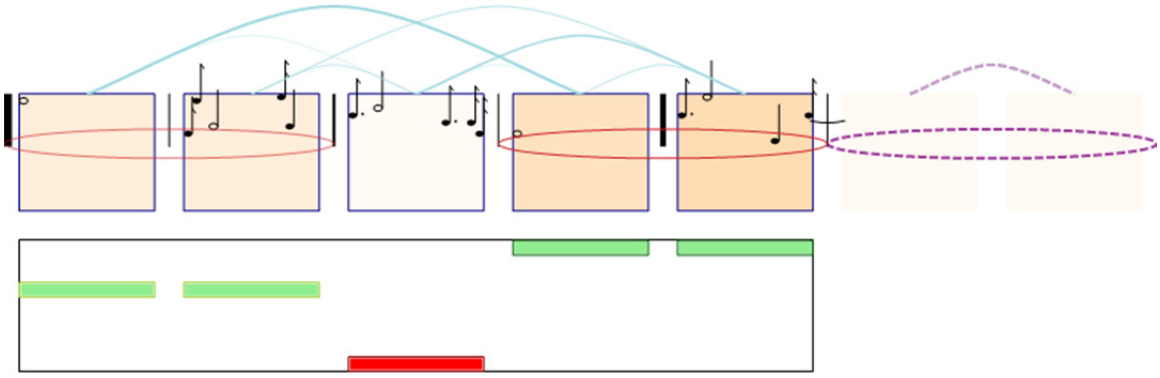

Figure 5.22: Bad Melody #1, measure 5.

One measure later, the picture has changed radically. Groups (1–2) and (4–5) are now in the picture, leaving an ungrouped measure 3 in the middle. Neither group is very strong. The strongest relationship found is the measure 1 to measure 4 connection found earlier. A new connection between measures 3 and 5 is also significant; the two measures start with the same rhythm.

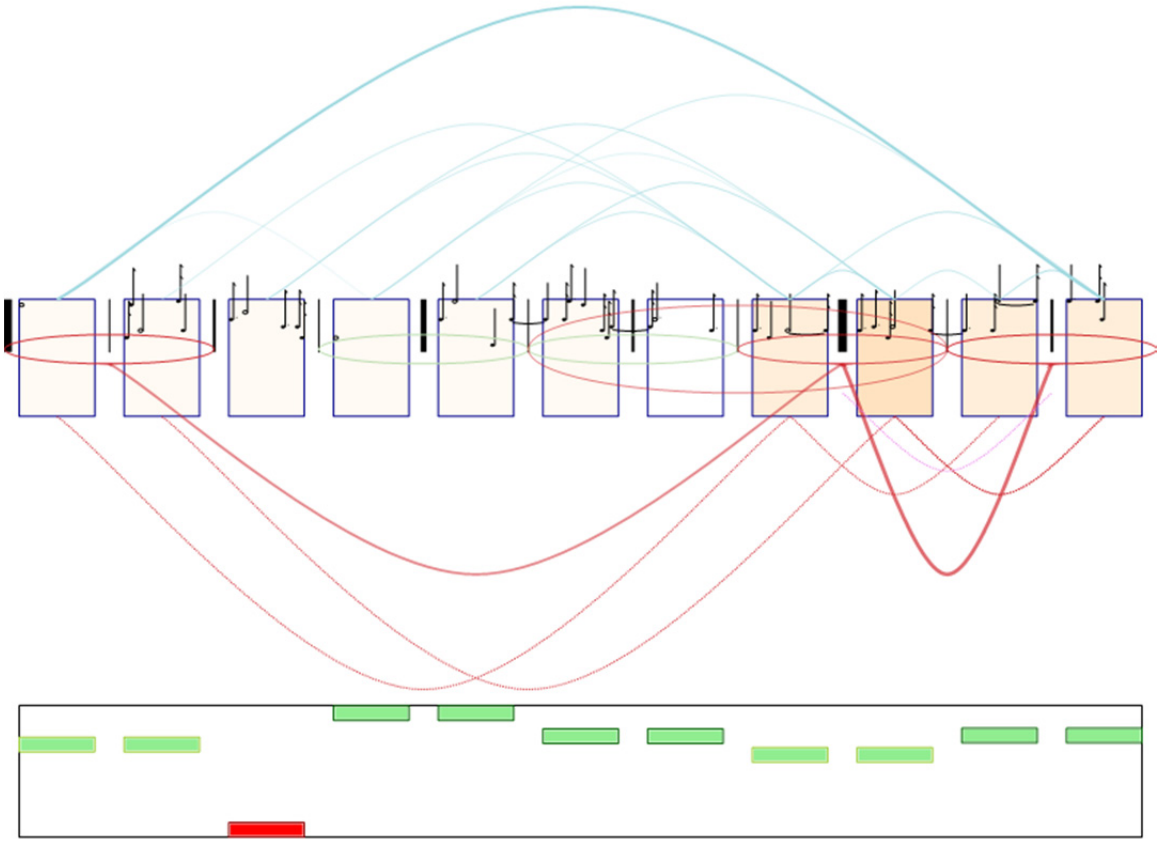

Figure 5.23: Bad Melody #1, end of processing.

Jumping ahead to the end of the listening performance, we see that the two groups present just after measure 5 was heard have survived in the program's representation. Likewise, measure 3 remains ungrouped. Several more groups have been found, as well as two analogies. It is hard to make sense of why the program has created these structures. Is this a bad listening performance? Or simply a good listening performance for a bad melody?

It turns out that the structures that have been found are quite weak. Musicat has a feature of the user interface — the "detail slider" — that allows the user to adjust the detail level up or down. By default, the detail level is set to the highest value and *all* structures are displayed. When the detail level is reduced, however, only the strongest structures are

displayed. The user can adjust the level to see which structures are really important in the program's representation.

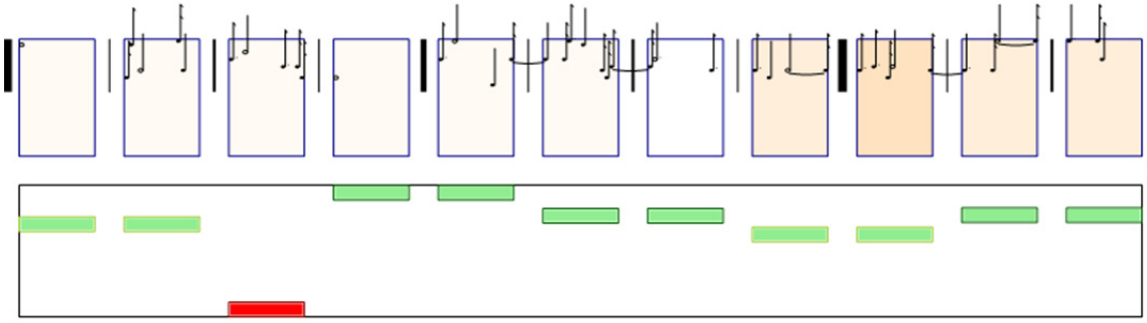

Figure 5.24: Bad Melody #1, low detail level.

When I lowered the detail level at the end of this run, every single one of the groups, links, and analogies disappeared. (Adjusting the detail level to a low-enough value will make all structures disappear for *any* run of Musicat on any melody, but in this case they all disappeared at the level that I typically use to focus attention on *strong* structures; I call this level the "low" detail level in the rest of the chapter.) Thus, *all* of the structures in this run were weak. This is as we would expect for a "bad melody". When listening to random notes, people naturally try to make sense of the notes and attempt to pick out some structure from the randomness; Musicat seems to be doing the same thing.

In the previous figure, the heights of the happiness bars may come as a surprise: they may seem higher than expected for such a (non-)melody. This is simply an artifact of how Musicat handles working memory: groups in the distant past are made stronger so that the program avoids modifying old groups. Once the melody is complete, most of the groups are in the distant past, and thus most groups in the Workspace have a high strength value, leading to high happiness values. However, either the presence of weak analogies or a complete lack of grouping structure will reduce happiness, even for old measures (for

example, in the figure, measures 1–2 are part of a weak analogy, and measure 3 is not a member of any group, so they have low happiness values.) See Chapter 7 for more details.

Raising the detail slider to a medium setting results in the following display (the happiness bars have been hidden in this figure, as well as in analogous figures later in this chapter, because happiness values are the same as in the previous figure, and the detail slider does not affect happiness):

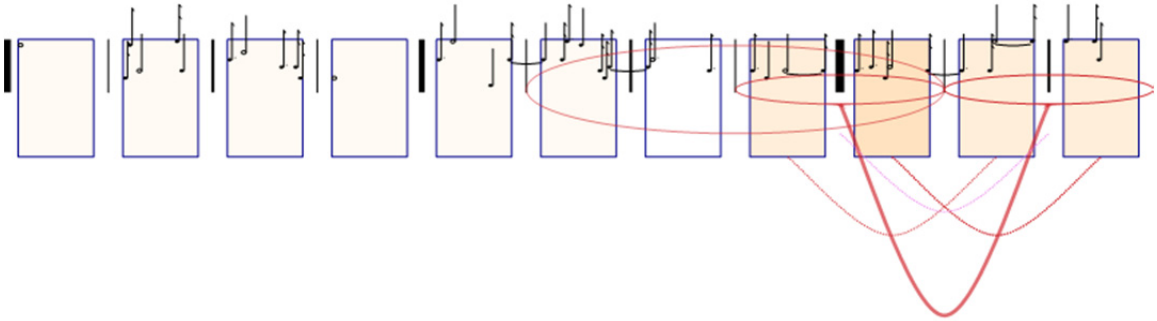

Figure 5.25: Bad Melody #1, medium level of detail.

How should one interpret this picture? These structures all disappeared under the "low" detail setting, so they aren't extremely important to how Musicat heard the melody. Still, let's examine what they might mean. Group **(6–9)** is hard to interpret. Recall that in the high-detail picture, there is a green group **(6–7)** inside the larger group, so this four-measure group may just be the simple result of joining the two smaller groups **(6–7)** and **(8–9)**, without very many other motivating reasons for the group to exist.

The analogy **(8–9)↔(10–11)** shows three supporting reasons for its existence. The dotted pink line just below the groups involved indicates a pitch-based relationship between the two groups, although it's not obvious from the musical score what the relationship might be. The two light-red arcs joining measure 8 to measure 10 and measure 9 to measure 11

indicate relationships Musicat discovered between these measures that support the analogy. However, it is also not obvious what these relationships are based on. Overall, Musicat's listening does not seem to tell us much about this melody — as expected, since the melody was random.

Bad Melody #2

The previous melody was so random that it was hard for us human listeners to make any sense of it at all. Since Musicat is highly sensitive to rhythmic patterns, I decided to make a new random melody with less variation in the rhythms. Specifically, I restricted the rhythm generator by removing dotted rhythms and making it more likely for quarter notes and eighth notes to fall on non-syncopated spots in a measure. I also made the sixteenth notes show up in more regular spots: they are required to appear in tandem with another sixteenth note within the space of quarter note, in such a way that it is impossible for later rhythms to start in places that are displaced by a single sixteenth note from a strong beat (*e.g.,* a rhythm such as sixteenth-quarter-quarter-quarter, starting on a downbeat, cannot occur). Finally, I associated selection weights to each rhythm so that some patterns (such as individual quarter notes) were more likely than others to occur. Pitch generation remained the same: notes were chosen randomly from the eight notes making up a C-major scale. The result of the first run of this new melody-generation algorithm is shown in the following figure.

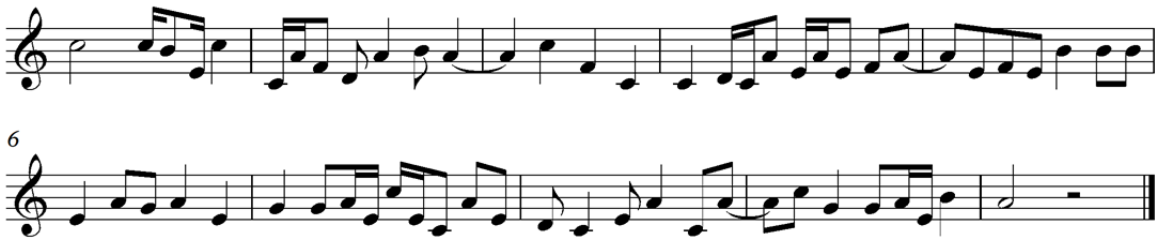

Figure 5.26: Bad Melody #2 (constrained rhythms).

This melody still sounds quite random to my ear, but much less so than Bad Melody #1. The melody seems to gain some internal coherence by virtue of the more regular generative process used to create the rhythm. Even though this process is random, the rhythmic vocabulary is so highly constrained that something is internally consistent about the rhythmic style. A similar statement is true for the pitch pattern of the melody; the pitches are random but are constrained to eight white piano keys. If an accidental were added to a single note of this melody, it would likely sound quite out of place.

Much to my surprise, when I played this "bad" melody for Douglas Hofstadter, he liked certain parts of it quite a lot, and even commented that it would be "a perfect theme for a Shostakovich fugue!" It reminded him in some way of Fugue 16 from Book 2 of Shostakovich's 24 Preludes and Fugues. A few happy coincidences served to make this melody better than either of us expected. It starts on the note C and then the note C appears several times in the first two measures, helping to establish C as the tonic. (Since C appears two times in the set of possible notes {C,D,E,F,G,A,B,C′}, C does have a higher probability of selection than other pitch classes.) As the melody progresses, the random pitches give the melody a more pandiatonic[6] character, but in the final two measures, the note A features prominently and it sounds as if the melody has modulated to A-minor.

[6] *Pandiatonic* melodies are ones that use the *notes* of a major scale, but without regard for the tonal *functions* typically implied by the scale.

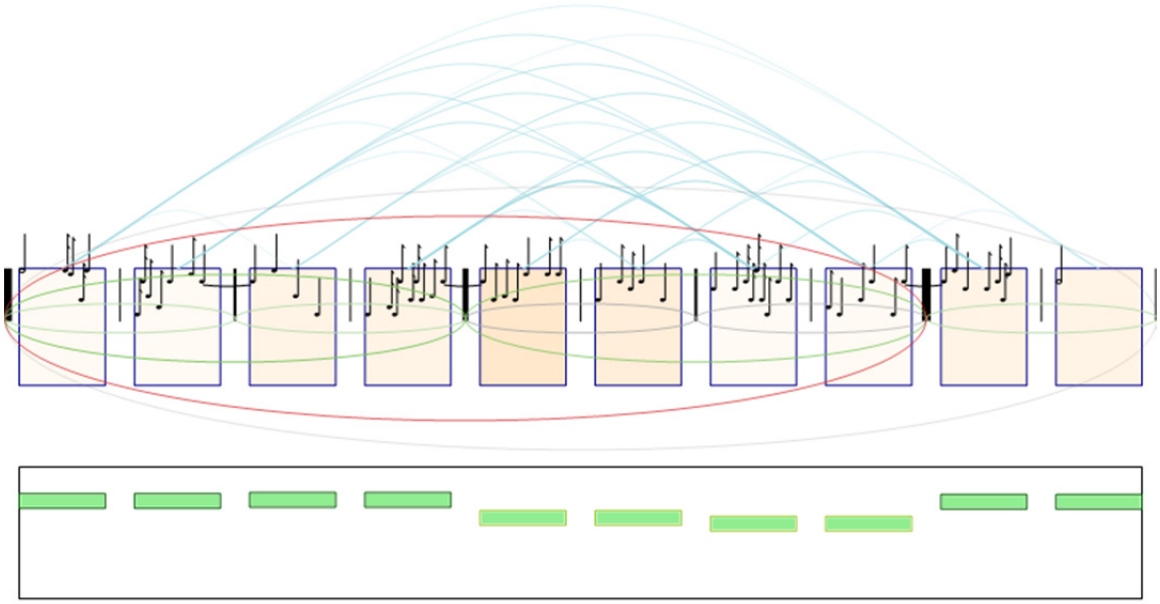

Figure 5.27: Bad Melody #2 (run 1), end of processing.

The previous figure shows the result of Musicat listening to the whole melody. (For the remainder of this chapter, I show intermediate states of the program running only when it seems particularly informative.) This listening performance has resulted in a more coherent structure than that for Bad Melody #1. Indeed, the first eight measures have been grouped in the standard binary fashion mentioned at the start of this chapter. This grouping reflects Musicat's bias to hear groups of measures in such a way that the lengths of the groups are powers of two, and pairs of adjacent groups are combined to form larger groups. In this example, the melody was 10 measures long, and we see that the final group **(9–10)** seems to have been tacked on somewhat arbitrarily onto the end, and included in the (very weak) meta-group **(1–10)** that spans all ten measures of the piece.

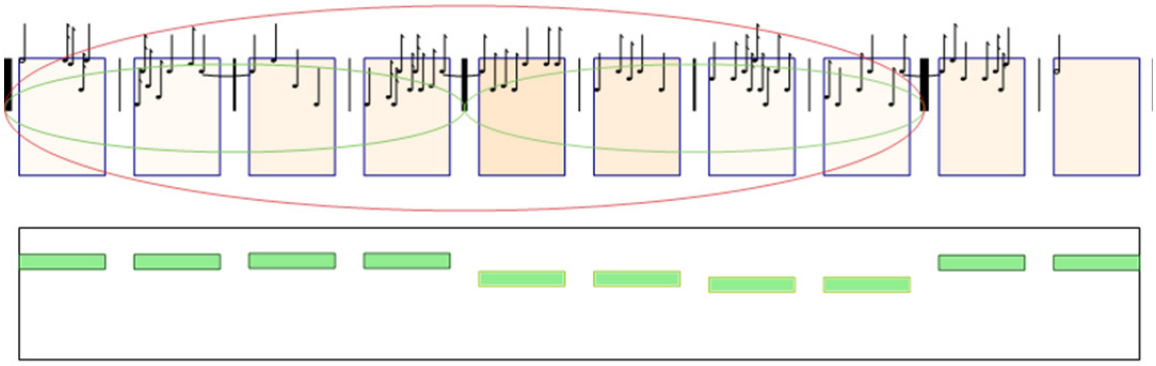

Figure 5.28: Bad Melody #2 (run 1), low level of detail.

With Bad Melody #1, turning down the detail level made all the groups disappear because they were so weak. In this case, by contrast, turning detail down to "low" leaves the two 4-measure groups and the 8-measure meta-group containing them still visible, while making all the 2-measure groups as well as the all-encompassing meta-group **(1–10)** disappear.

Since Musicat is nondeterministic, it was certain that different groups would form on different runs of the program. I therefore tried running it twice more; the results follow.

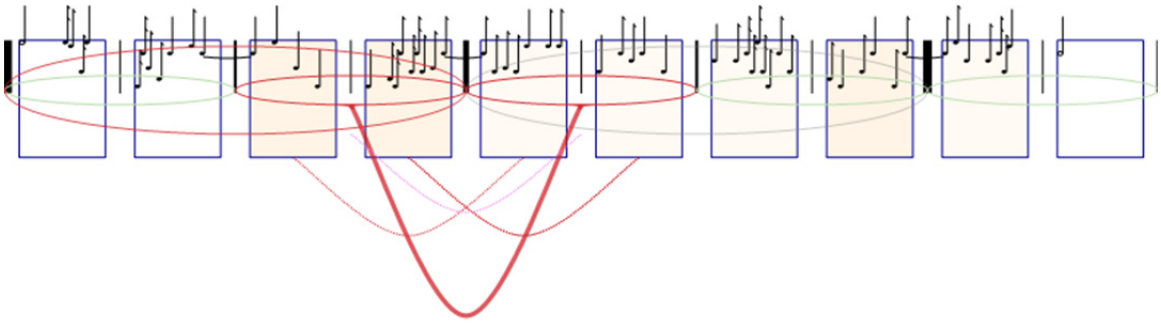

Figure 5.29: Bad Melody #2 (run 2), low level of detail.

The structures formed during this second run are similar to those in the previous run, although now five 2-measure groups are still visible even in low-detail mode. Also, an analogy

was formed this time. The analogy may be partly due to a weak melody-contour relationship found between the two groups concerned.

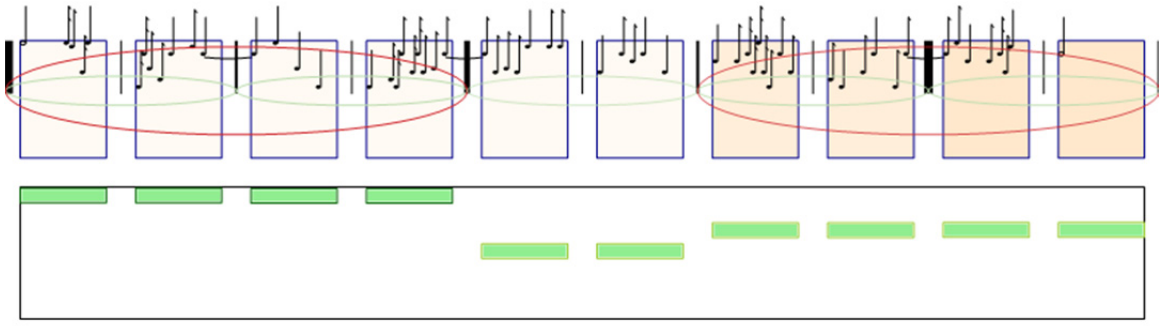

Figure 5.30: Bad Melody #2 (run 3), low level of detail.

In a third run, the grouping structure turned out a little different: the 4-measure and 2-measure groups in the final 6 measures have switched places, resulting in a symmetric grouping structure for the melody. As in the first run, no analogies were found.

Even if I didn't know about the algorithms that generated these two bad melodies, it is obvious just from observing the difference in Musicat's grouping structures that there was an important difference between the melodies. Musicat was clearly responding to some quality of this melody that was not present in Bad Melody #1. The randomness of these melodies resulted in little analogy-formation activity. But in this second bad melody, the more-structured (but still random) rhythms allowed the program's binary-structure default to become apparent in the perceived grouping structure in each of the three runs.

Bad Melody #3

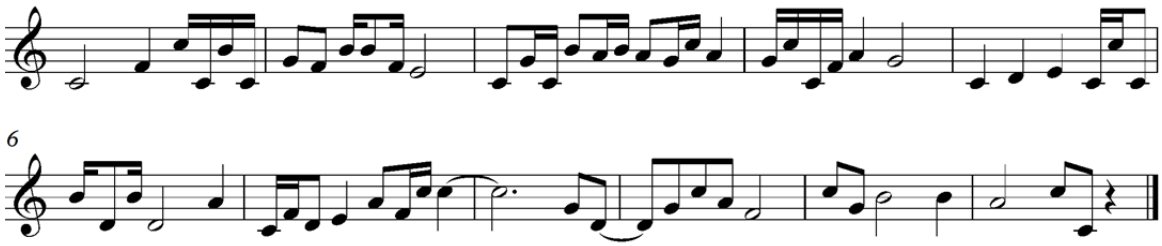

Figure 5.31: Bad Melody #3 (constrained rhythm).

I generated Bad Melody #3 using the same program that generated Bad Melody #2; the only difference was the choice of random seed. This melody makes a bit less sense to me than the previous one (although, amusingly, at the start of measure 5 the opening of the melody *Frère Jacques* appears, quite by accident— *Frère Jacques* appears on purpose in the next chapter as one of the Simple Melodies).

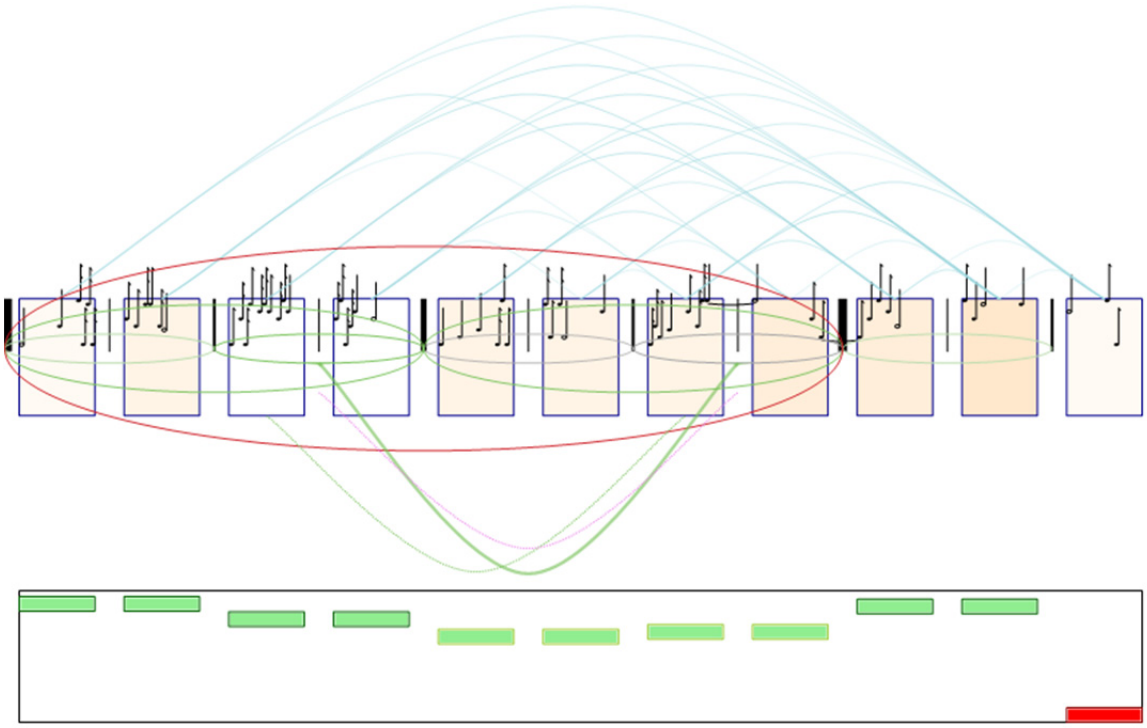

Figure 5.32: Bad Melody #3 (run 1).

Does Musicat "hear" this melody any differently than the previous one? The groups are nearly the same. This melody is one measure longer, and measure 11 has not been included in any group. An analogy has formed, **(3–4)↔(7–8)**, supported by a contour relationship as well as by a rhythmic relationship between measures 3 and 7. However, the analogy is not very strong; in the low-detail view it is not visible:

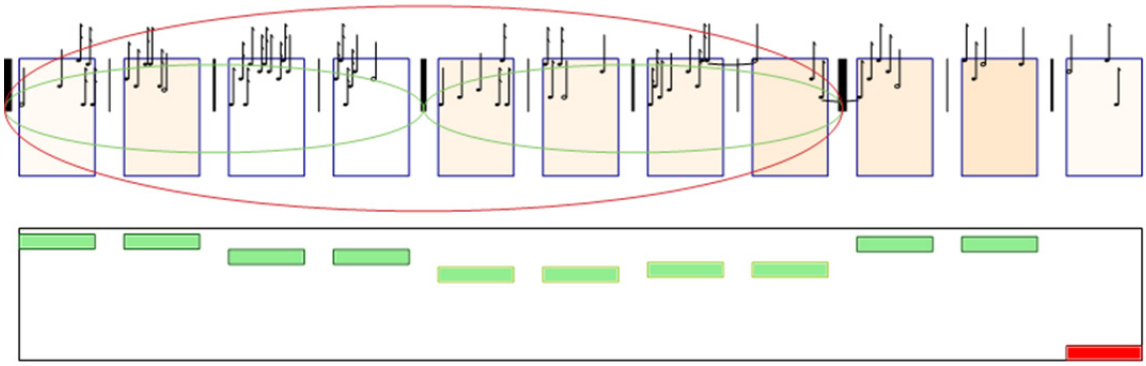

Figure 5.33: Bad Melody #3 (run 1), low detail.

This picture in the diagram above looks very similar to two of the pictures for Bad Melody #2. Analogous 4- and 8-measure groups show up here.

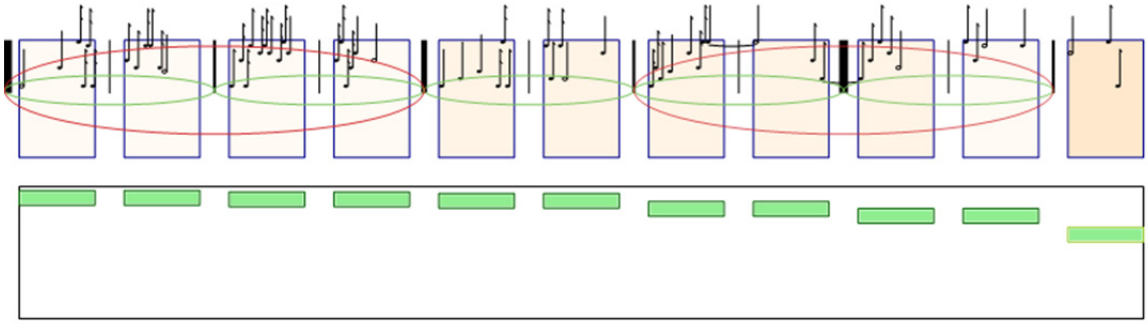

Figure 5.34: Bad Melody 3 (run 2), low detail.

Similarly, in a second run, we find the alternate symmetric grouping structure that we encountered in Bad Melody #2 (run 3), and again no analogy has formed.

Musicat attempts to find structure in these random melodies, resulting in these characteristic patterns. It is a bit surprising that it "hears" groups where we might not expect any, but it's possible that human listeners would also hear structures such as these when given these random melodies. Less surprising, however, is that strong analogies were not formed

frequently. In the next chapter, we will see that for typical melodies, Musicat finds many more analogies, and they are often very strong.

Bad Melody #4

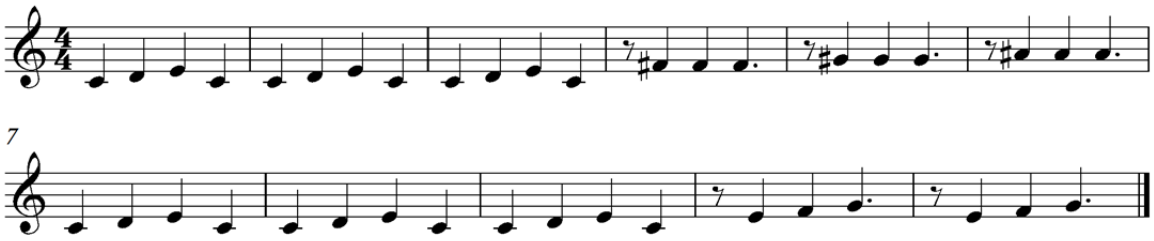

Figure 5.35: Bad Melody #4 (odd-length phrases).

This melody was not composed randomly, and unlike the others in this section, it is not *bad* in the same sense of sounding very disjointed. Instead, I myself deliberately created it to have group boundaries that occur in surprising places but that should be obvious to a human listener. This is achieved by using very simple repetition to form groups, but then forcing the groups to have odd numbers of measures. Also, notice that this melody is a distortion of the *Frère Jacques* melody that appears later in this chapter. I would expect a human to hear the melody as having the following groups: **(1–3)**, **(4–6)**, **(7–9)**, **(10–11)**, and perhaps higher-level groups **(1–5)** and **(7–11)**. The 2- and 3-measure groups are particularly obvious because each one consists of either exact repetition, for groups **(1–3)**, **(7–9)**, and **(10–11)**, or a simple sequence (a repetition of each measure, with transposition), in the case of group **(4–6)**. Can Musicat overcome its bias favoring groups with lengths that are powers of 2?

For this melody it is illuminating to see what happens at intermediate stages in the listening performance.

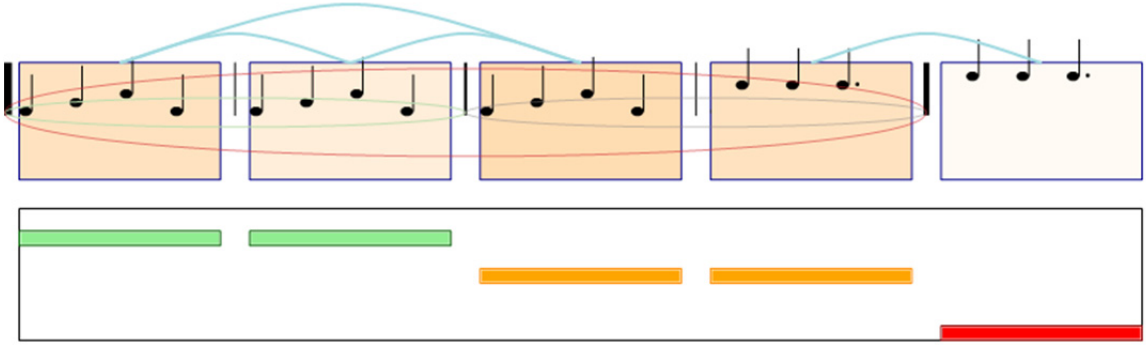

Figure 5.36: Bad Melody #4, measure 5.

After hearing measure 5, the program has created two 2-measure groups, as well as a meta-group **(1–4)** containing these two groups, so it seems as if the powers-of-2 bias is overpowering the natural grouping structure of this melody. However, the happiness rectangles show something interesting: whereas the first two measures are relatively happy (the happiness bars are green), the second two measures are less happy (orange bars). Group **(3–4)** is weak, resulting in the low happiness values. Also, the final measure has minimal happiness (red bar) because it is not included in any group.

Despite the questionable grouping structure displayed here, Musicat has noticed the obvious similarities in the measures, which are shown by the measure links above the measures. Notice that measures 1, 2, and 3 are all linked together, as are measures 4 and 5. All the clues are in place for Musicat to overcome its default pressure to form groups of lengths 2, 4, 8, 16, and so on.

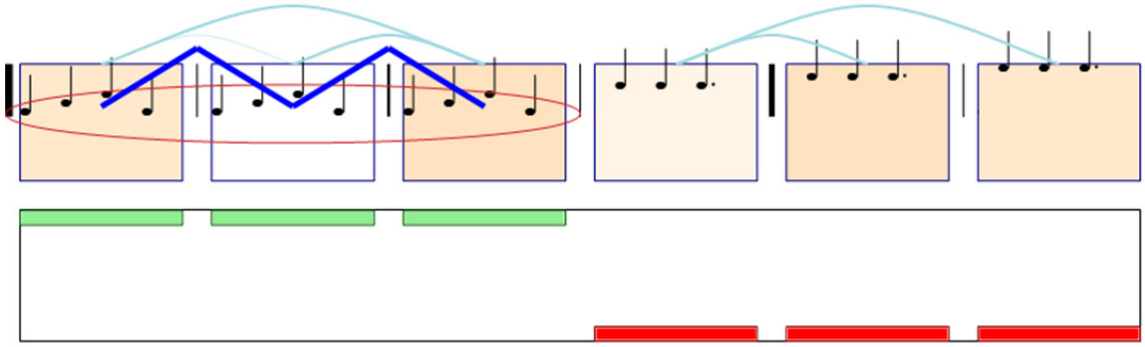

Figure 5.37: Bad Melody #4, measure 6.

After measure 6 has been heard, something radical has happened! A group of length 3 was indeed created as desired. The thick blue angled lines in the figure indicate that Musicat has created a *sequence* spanning measures 1–3. This came as a surprise to me, because Musicat's definition of "sequence" requires successive tonal transpositions between measures; typically a sequence would involve all the notes in each successive measure shifting up or down by a scale step. However, in Musicat there are no constraints on the size of the transposition, so it "heard" these measures as successive transpositions by 0 steps!! A measure followed by two copies of itself is, to be sure, a type of sequence, albeit a trivial one.

An additional measure link has formed to link measures 1 and 3, but the next expected group **(4–6)** has not yet formed.

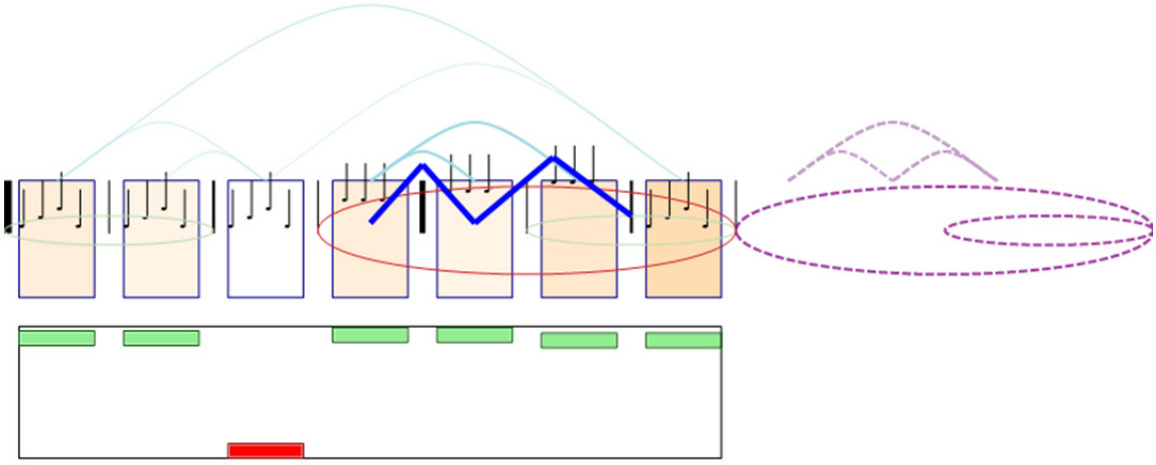

Figure 5.38: Bad Melody #4, measure 7.

After measure 7 is heard, we have both good and bad news. Another sequence has been formed, but the old sequence has disappeared, apparently having lost a competition to the initial **(1–2)** group, leaving measure 3 ungrouped and unhappy. The new sequence is, curiously, not made out of three measures, as would be expected. Instead, the third element of the sequence is the peculiar group **(6–7)**, so the sequence itself constitutes a 4-measure meta-group. This is explicitly allowed by Musicat's definition of sequence: the final element of the sequence is allowed to be a group that has extra material tacked on to the end, resulting in a group with longer duration than earlier elements of the sequence (this often happens at the end of sequences in traditional music: for example, a melodic pattern might appear in three successive measures, but the pattern in the final measure would be extended to fill out a 4-measure phrase). In this case, however, the group **(6–7)** is certainly not one we expected to form.

Another problem visible here is the thick bar line between measures 4 and 5. Ideally. the sequence starting with measure 4, in concert with the sequence ending right after measure 3, would cause a thick bar line to appear between measures 3 and 4, reinforcing that

spot as a place where groups should end or start. Unfortunately, though, in the figure, the thick bar line is in the "wrong" place and weakens the perceived strength of the initial sequence for Musicat, making the desired grouping structure less likely to form and become stable.

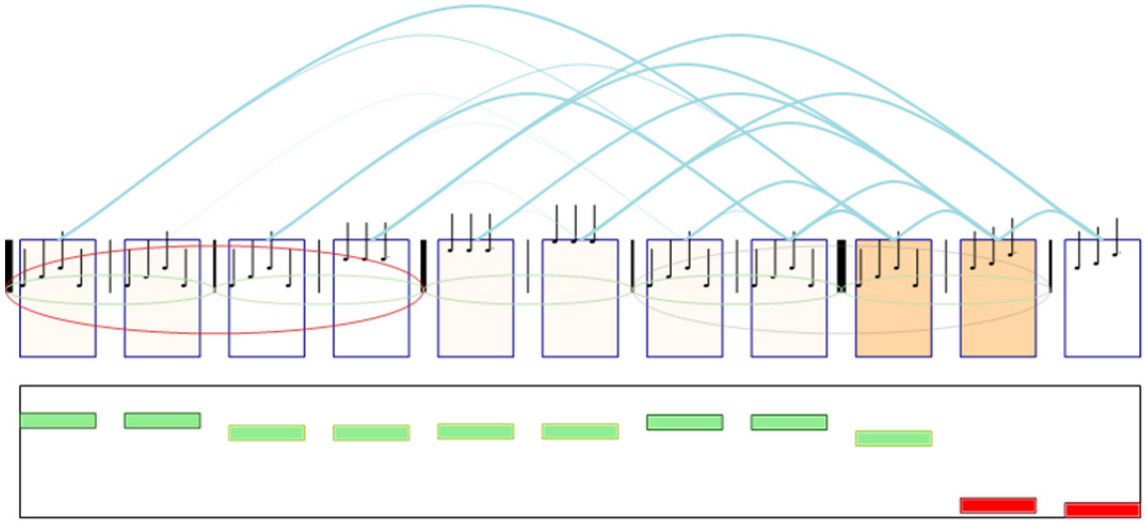

Figure 5.39: Bad Melody #4, end of processing.

After hearing all 11 measures, the program has reverted to a very stereotyped 2- and 4-measure group structure, unfortunately. The sequences did not persist through the run. I tried another run, with similar results, in which the hoped-for sequences would appear briefly, only to be destroyed by a rival length-2 group. The final grouping structure is again one of the familiar structures we saw in Bad Melodies #2 and #3. Also, no analogies were formed (although many strong measure links were discovered).

This run, though disappointing, is interesting because it gives a glimpse into the internal pressures that drive how Musicat listens. In this case, the pressure for length-2 and length-4 groups clearly overwhelmed the pressure to hear groups made up of exact repetitions

and sequences, but during the middle of the listening performance, it was not clear which pressures would prevail in the end.

Bad Melody #5

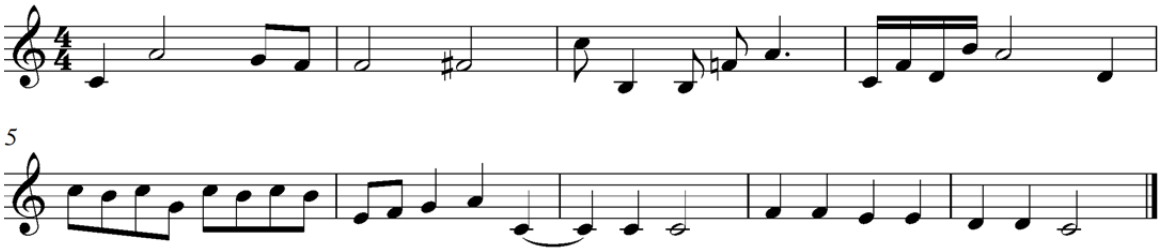

Figure 5.40: Bad Melody #5.

This melody starts with four rather arbitrary measures (measures 1 and 2 seem to go together, but measures 3 and 4 seem like independent musical thoughts. These are followed by several measures of other melodies, simply cut-and-pasted together. Measure 5 is from the start of "Good People All" (which appears in a later section), measures 6–7 are based on "Younger than Springtime", and the final two measures come from the end of "Twinkle, Twinkle".

Since this version of Musicat does not have a long-term memory that allows it to recall musical motifs from pieces it has heard in the past, it is not in the least influenced by the melodic allusions in this example. For a human listener, of course, such remindings would very likely influence the way this melody is heard, probably resulting in a sensation of amusement or even hilarity, based on the utter incongruity of the quotations' meaningless juxtaposition.

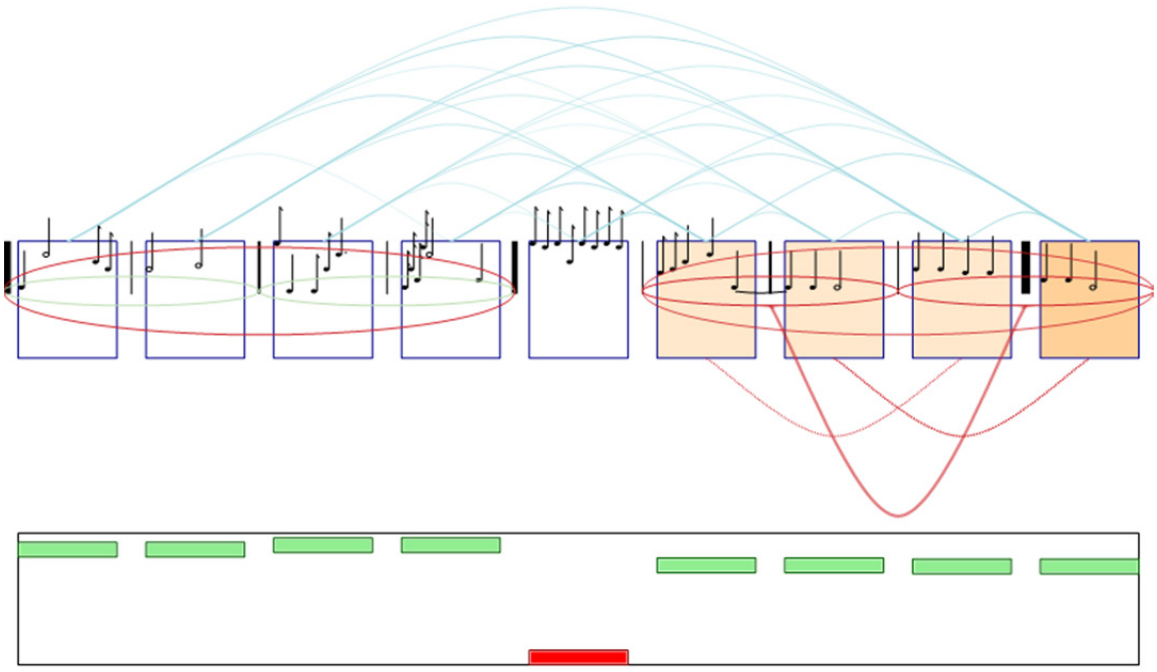

Figure 5.41: Bad Melody #5 (run 1).

The results of the first run on this melody surprised me. The program made an analogy that I didn't expect: **(6–7)↔(8–9)**. I didn't expect it because I had created the melody by arbitrarily pasting measures 6–7 from one melody and measures 8–9 from another. The analogy makes sense, though: the rhythm of measure 7 is very similar to that of measure 9 and they both end on the note C. The program has also found the not totally unreasonable group **(1–2)**, and group **(3–4)** also seems comprehensible. Most interestingly, measure 5 is not part of any group, despite the program's unhappiness with this situation (which would constitute a pressure to try to find a way to include it in a group). This seems reasonable; measure 5 is saliently different from all the other measures in the melody. Perhaps a more human-like listening would include this measure as the only member of a trivial 1-measure group. Overall, though, this listening performance by Musicat seems human-like in many ways.

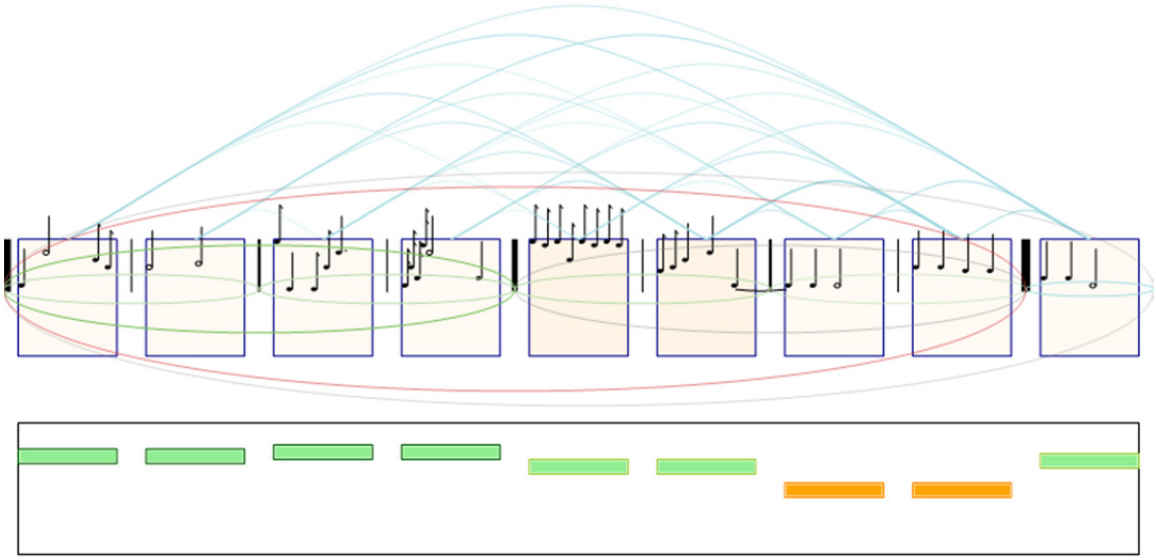

Figure 5.42: Bad Melody #5 (run 2).

On a second run on the same melody, the program found a different structure. It has simply grouped the first 8 measures in a regular binary fashion, with measure 9 tacked on at the end to make a large (but very weak) 9-measure group. Measures 7 and 8 have low happiness values, but they nonetheless form a group, even though the group **(8–9)** would have been much stronger. This run is less satisfying to me than the previous one was, because it doesn't seem to parallel a human listening experience nearly as well as that run did.

SUMMARY

Musicat gave mixed results on these five "bad melodies". In places its listening performance seems similar to a human's. Specifically, for Bad Melody #1 and Bad Melody #5 (run 1), it behaved essentially as I hoped. In Bad Melody #4, the program made some of the expected structures during the run, but disappointingly, they did not persist all the way to the end of the run. In the other examples, the program exhibited a preference for hearing melodies in terms of a regular binary structure, but it is highly questionable whether such structure was warranted for these random melodies.

Even though Musicat made some *groups* that humans might not have created, in most of these examples, the program did not find many strong *analogies* between groups. Such a lack of analogies is to be expected for random melodies. In the next chapter, by contrast, we will see how Musicat typically finds *many* analogies in simple melodies; its listening performances for random melodies will (fortunately) turn out to be very different from its listening performances for more typical, coherent melodies.